The Data Science Model

Math 396

Treena Basu

Department of Mathematics
Occidental College

February 9, 2021

Summary

What is Data Science?

Data Science

Data science is a broad field that concerns itself with the extraction of knowledge and insights from data.

Data comes in different shapes and forms.

Data Science

Data science is a broad field that concerns itself with the extraction of knowledge and insights from data.

Data comes in different shapes and forms.

What is Machine Learning?

Machine Learning

Machine Learning (ML) is an application of artificial intelligence (AI) that focuses on the creation and development of mathematical models that use data to **learn** for themselves.

The primary motivation behind creating these mathematical models or ML algorithms is to allow the model to **learn** automatically without human intervention or assistance and adjust actions accordingly.

Machine Learning

Machine Learning (ML) is an application of artificial intelligence (AI) that focuses on the creation and development of mathematical models that use data to **learn** for themselves.

The primary motivation behind creating these mathematical models or ML algorithms is to allow the model to **learn** automatically without human intervention or assistance and adjust actions accordingly.

Types of Machine Learning

Machine learning approaches are traditionally divided into three broad categories:

- Supervised Learning: the model is "trained" with example inputs and their desired outputs; the goal is to learn a general rule that maps inputs to outputs
- Unsupervised Learning: the model is "trained" with unlabeled data, leaving it on its own to find structure in its input; the goal is to discover hidden patterns in data
- Reinforcement Learning: the model is receives "training" in the form of feedback that's analogous to rewards, which it tries to maximize

Types of Machine Learning

Machine learning approaches are traditionally divided into three broad categories:

- Supervised Learning: the model is "trained" with example inputs and their desired outputs; the goal is to learn a general rule that maps inputs to outputs
- Unsupervised Learning: the model is "trained" with unlabeled data, leaving it on its own to find structure in its input; the goal is to discover hidden patterns in data
- Reinforcement Learning: the model is receives "training" in the form of feedback that's analogous to rewards, which it tries to maximize

Types of Machine Learning

Machine learning approaches are traditionally divided into three broad categories:

- Supervised Learning: the model is "trained" with example inputs and their desired outputs; the goal is to learn a general rule that maps inputs to outputs
- Unsupervised Learning: the model is "trained" with unlabeled data, leaving it on its own to find structure in its input; the goal is to discover hidden patterns in data
- Reinforcement Learning: the model is receives "training" in the form of feedback that's analogous to rewards, which it tries to maximize

Types of Machine Learning

Machine learning approaches are traditionally divided into three broad categories:

- Supervised Learning: the model is "trained" with example inputs and their desired outputs; the goal is to learn a general rule that maps inputs to outputs
- Unsupervised Learning: the model is "trained" with unlabeled data, leaving it on its own to find structure in its input; the goal is to discover hidden patterns in data
- Reinforcement Learning: the model is receives "training" in the form of feedback that's analogous to rewards, which it tries to maximize

Supervised Learning

In a supervised learning problem we are given a set of data $(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$.

The goal of the model is to be able to learn a function $f(x)$ to predict y given x.

Note:

- x can be multi-dimensional where each dimension corresponds to an attribute.
- y can be real valued (Regression Problem) or y can be categorical (Classification Problem).

Supervised Learning

In a supervised learning problem we are given a set of data $(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$.

The goal of the model is to be able to learn a function $f(x)$ to predict y given x.

Note:

- x can be multi-dimensional where each dimension corresponds to an attribute.
- y can be real valued (Regression Problem) or y can be categorical (Classification Problem).

Supervised Learning

In a supervised learning problem we are given a set of data $(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$.

The goal of the model is to be able to learn a function $f(x)$ to predict y given x.

Note:

- x can be multi-dimensional where each dimension corresponds to an attribute.
- y can be real valued (Regression Problem) or y can be categorical (Classification Problem).

Supervised Learning

In a supervised learning problem we are given a set of data $(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$.

The goal of the model is to be able to learn a function $f(x)$ to predict y given x.

Note:

- x can be multi-dimensional where each dimension corresponds to an attribute.
- y can be real valued (Regression Problem) or y can be categorical (Classification Problem).

Supervised Learning

In a supervised learning problem we are given a set of data $(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$.

The goal of the model is to be able to learn a function $f(x)$ to predict y given x.

Note:

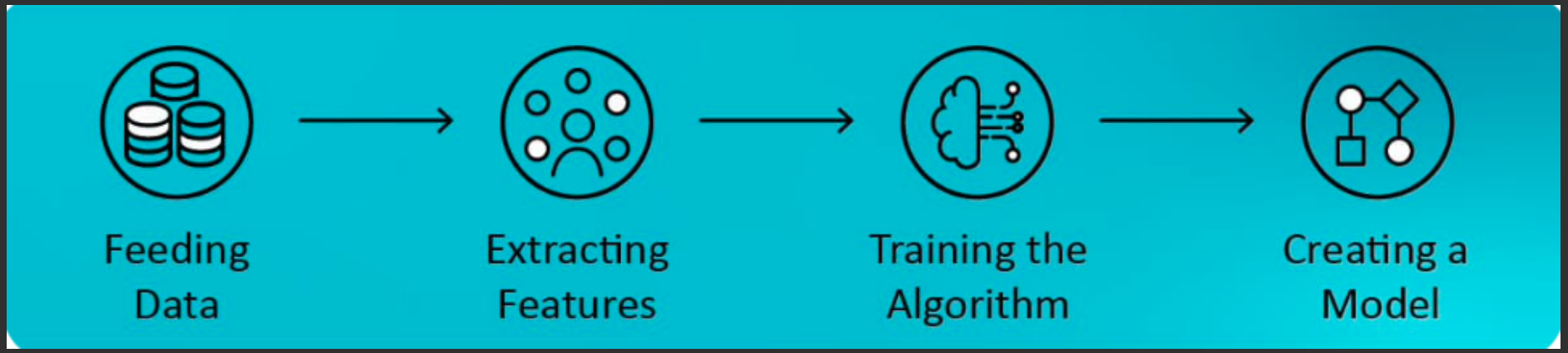
- x can be multi-dimensional where each dimension corresponds to an attribute.
- y can be real valued (Regression Problem) or y can be categorical (Classification Problem).

Examples of Supervised Learning

- **Predicting House Prices** (Regression Problem)

  We will leverage data coming from thousands of houses, their attributes (inputs: location, square footage, number of bedrooms, etc) and prices (labels or outputs), and train a supervised machine learning model to predict a new house's price based on the examples observed by the model.
- **Is it a Cat or Dog?** (Classification Problem) We train a supervised ML algorithm using labelled pictures of cats and dogs. We then give the model an image it has never seen before with goal that the model will predict what class the image belongs to: cat or dog.

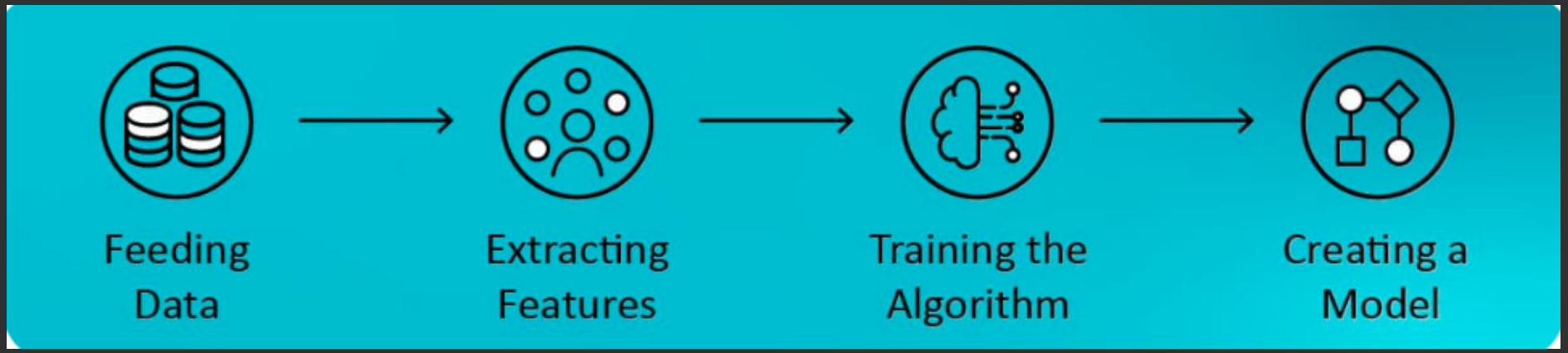
Components of a ML Algorithm



Every ML algorithm has the following three components:

- Mathematical Representation
- Parameter Optimization
- Performance Evaluation

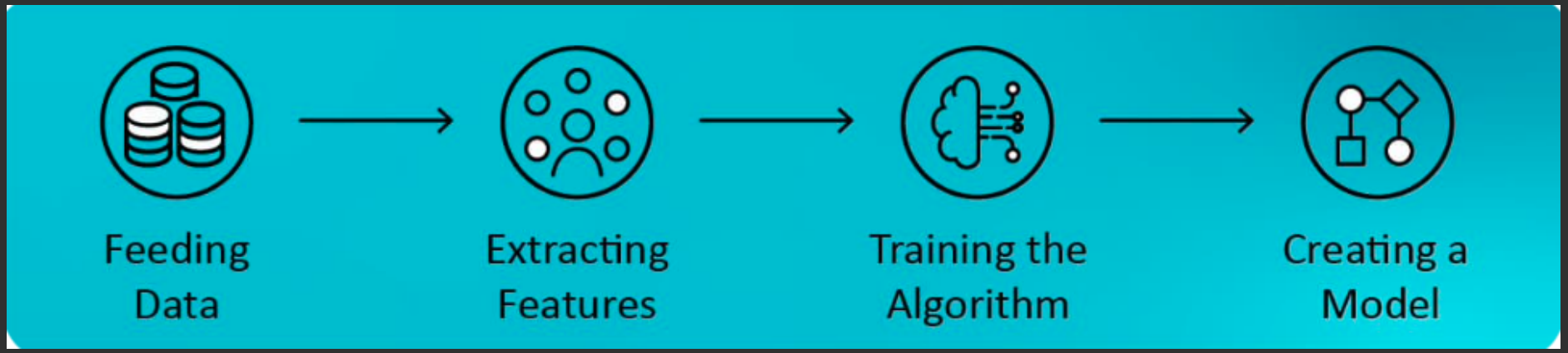
Components of a ML Algorithm



Every ML algorithm has the following three components:

- Mathematical Representation
- Parameter Optimization
- Performance Evaluation

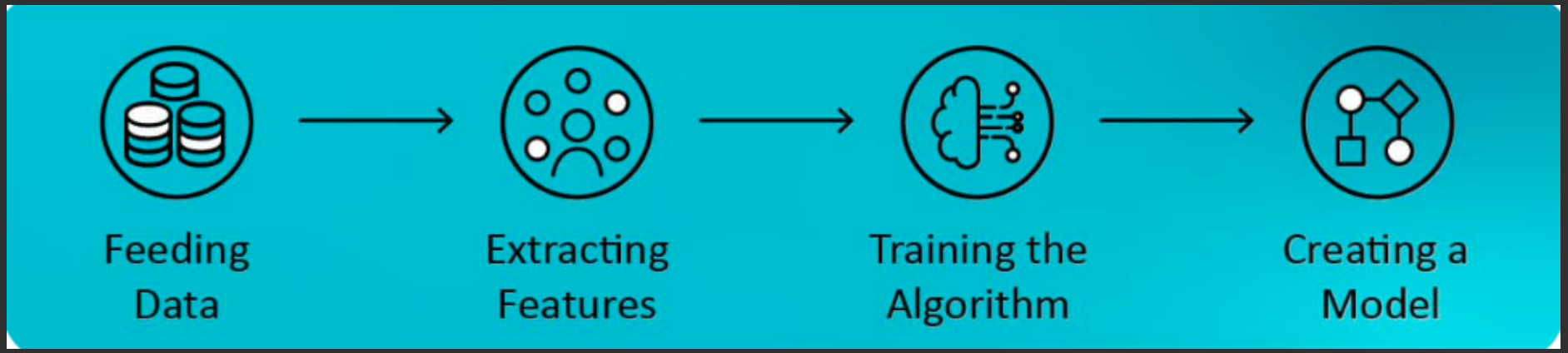
Components of a ML Algorithm



Every ML algorithm has the following three components:

- Mathematical Representation
- Parameter Optimization
- Performance Evaluation

Components of a ML Algorithm



Every ML algorithm has the following three components:

- Mathematical Representation
- Parameter Optimization
- Performance Evaluation

Some Supervised ML Algorithms for Classification

Listed below are some standard algorithms designed for classification problems:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines (SVM)
- Naive Bayes
- K Nearest Neighbors (KNN)

Each algorithm has its own set of parameters that can be optimized for better model performance.

Some Supervised ML Algorithms for Classification

Listed below are some standard algorithms designed for classification problems:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines (SVM)
- Naive Bayes
- K Nearest Neighbors (KNN)

Each algorithm has its own set of parameters that can be optimized for better model performance.

Some Supervised ML Algorithms for Classification

Listed below are some standard algorithms designed for classification problems:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines (SVM)
- Naive Bayes
- K Nearest Neighbors (KNN)

Each algorithm has its own set of parameters that can be optimized for better model performance.

Some Supervised ML Algorithms for Classification

Listed below are some standard algorithms designed for classification problems:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines (SVM)
- Naive Bayes
- K Nearest Neighbors (KNN)

Each algorithm has its own set of parameters that can be optimized for better model performance.

Some Supervised ML Algorithms for Classification

Listed below are some standard algorithms designed for classification problems:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines (SVM)
- Naive Bayes
- K Nearest Neighbors (KNN)

Each algorithm has its own set of parameters that can be optimized for better model performance.

Some Supervised ML Algorithms for Classification

Listed below are some standard algorithms designed for classification problems:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines (SVM)
- Naive Bayes
- K Nearest Neighbors (KNN)

Each algorithm has its own set of parameters that can be optimized for better model performance.

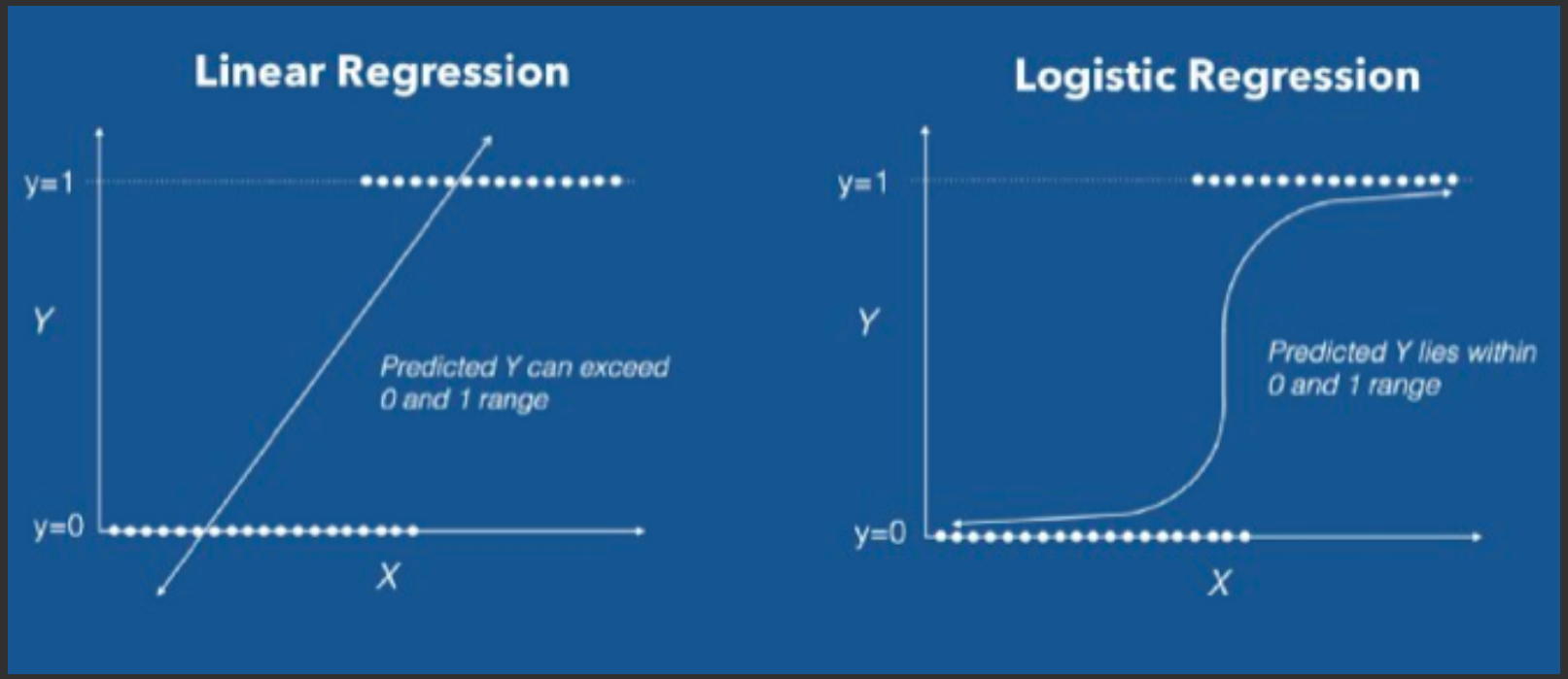
Some Supervised ML Algorithms for Classification

Listed below are some standard algorithms designed for classification problems:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines (SVM)
- Naive Bayes
- K Nearest Neighbors (KNN)

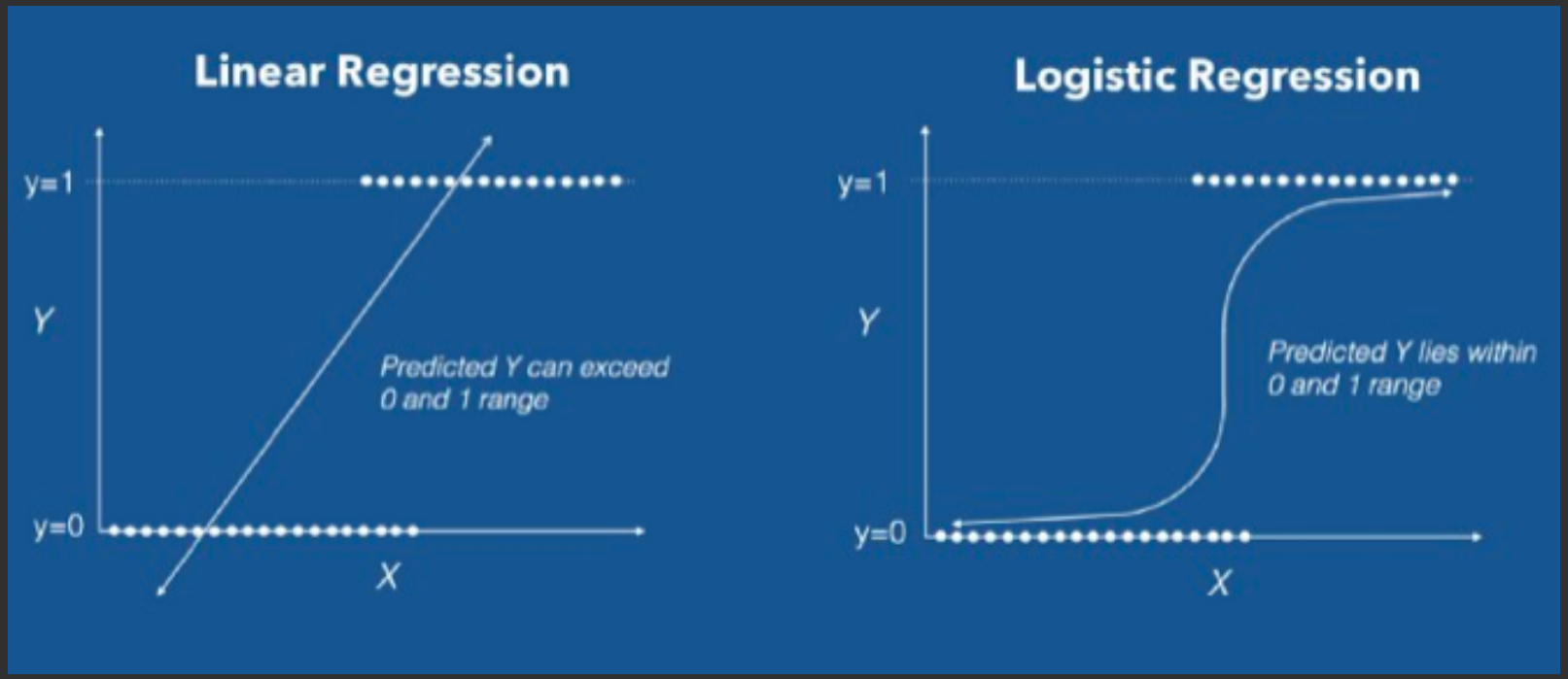
Each algorithm has its own set of parameters that can be optimized for better model performance.

Linear Regression Versus Logistic Regression Visually



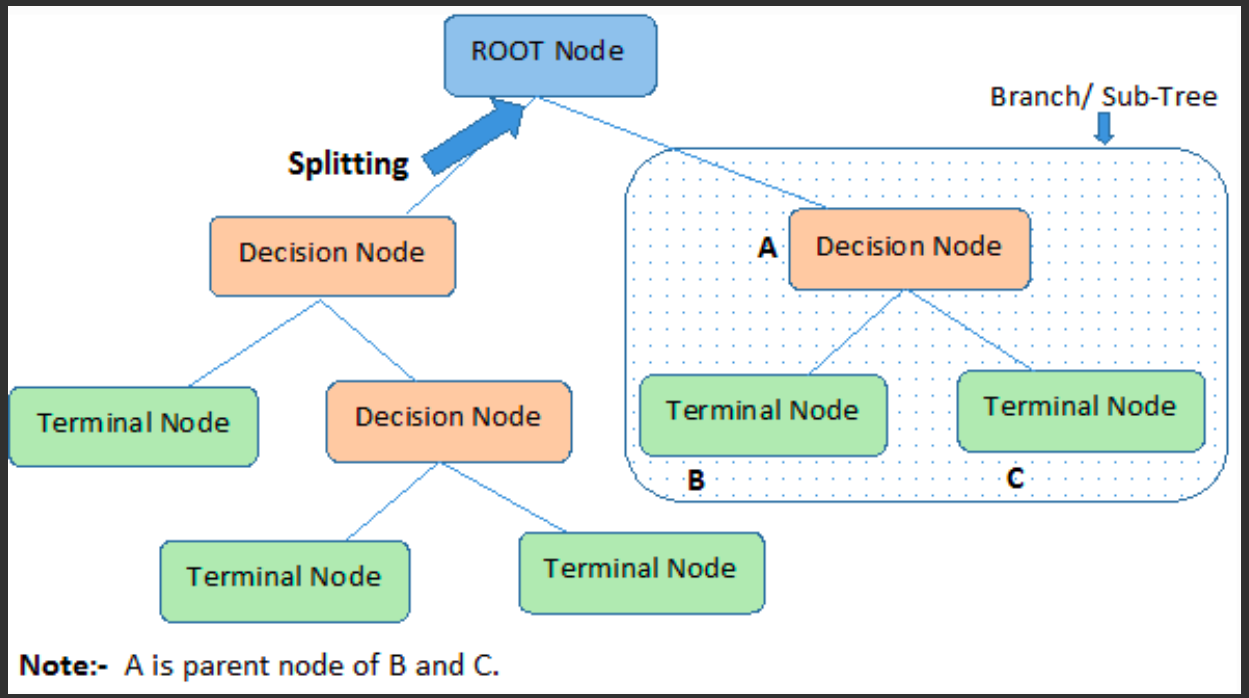
Logistics Regression uses a method called Gradient Descent to optimize parameters in the Log Loss Function.

Linear Regression Versus Logistic Regression Visually



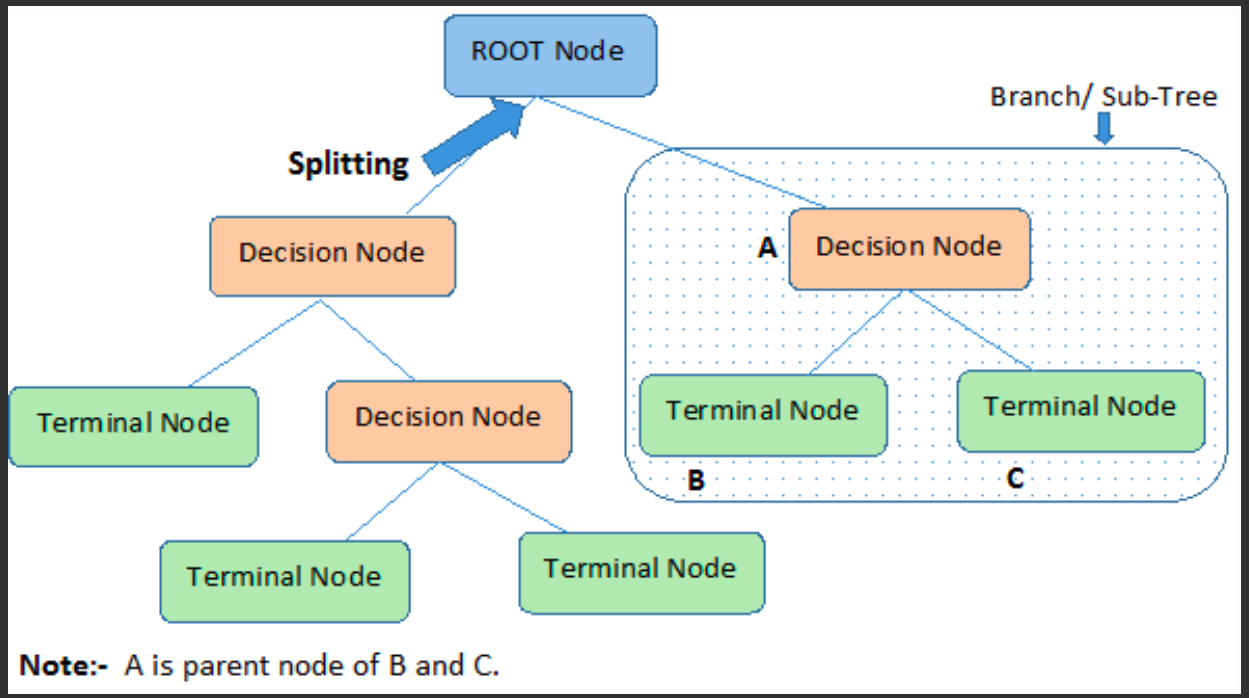
Logistics Regression uses a method called Gradient Descent to optimize parameters in the Log Loss Function.

Decision Trees



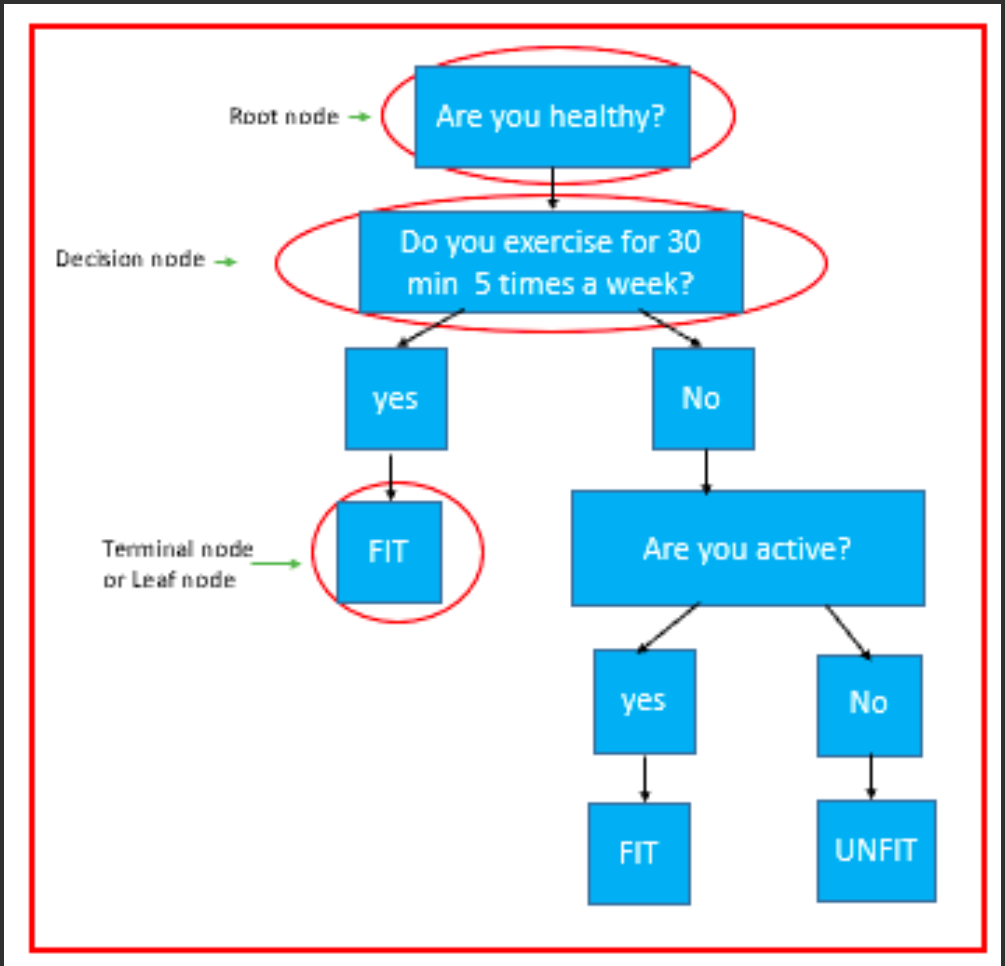
Decision Trees uses a measure called entropy to find a way to split the set until all the data belongs to one of two classes (in case of binary classification).

Decision Trees



Decision Trees uses a measure called entropy to find a way to split the set until all the data belongs to one of two classes (in case of binary classification).

Decision Trees



Classification Problems
Fraud Detection

Credit Card Fraud Detection

Consider a data set that documents a total of **284,807** credit card transactions of which **only 492** are fraudulent and the rest (284,315) are considered normal transactions.

We would like to build **(train)** a model that will be able to successfully **classify** future transactions as either normal or fraudulent, with the intention of immediately stopping a fraudulent transaction from being completed.

Since we are interested in identifying future fraudulent transactions we will label fraudulent transactions as the **positive class** and normal transactions as the **negative class**.

The data set is highly **unbalanced**, the positive class (frauds) account for 0.172% of all transactions.

Credit Card Fraud Detection

Consider a data set that documents a total of **284,807** credit card transactions of which **only 492** are fraudulent and the rest (284,315) are considered normal transactions.

We would like to build **(train)** a model that will be able to successfully **classify** future transactions as either normal or fraudulent, with the intention of immediately stopping a fraudulent transaction from being completed.

Since we are interested in identifying future fraudulent transactions we will label fraudulent transactions as the **positive class** and normal transactions as the **negative class**.

The data set is highly **unbalanced**, the positive class (frauds) account for 0.172% of all transactions.

Credit Card Fraud Detection

Consider a data set that documents a total of **284,807** credit card transactions of which **only 492** are fraudulent and the rest (284,315) are considered normal transactions.

We would like to build **(train)** a model that will be able to successfully **classify** future transactions as either normal or fraudulent, with the intention of immediately stopping a fraudulent transaction from being completed.

Since we are interested in identifying future fraudulent transactions we will label fraudulent transactions as the **positive class** and normal transactions as the **negative class**.

The data set is highly **unbalanced**, the positive class (frauds) account for 0.172% of all transactions.

Credit Card Fraud Detection

Consider a data set that documents a total of **284,807** credit card transactions of which **only 492** are fraudulent and the rest (284,315) are considered normal transactions.

We would like to build **(train)** a model that will be able to successfully **classify** future transactions as either normal or fraudulent, with the intention of immediately stopping a fraudulent transaction from being completed.

Since we are interested in identifying future fraudulent transactions we will label fraudulent transactions as the **positive class** and normal transactions as the **negative class**.

The data set is highly **unbalanced**, the positive class (frauds) account for 0.172% of all transactions.

Steps to Developing a Model

In order to develop a classifier that will identify transactions as either fraudulent or normal we perform the following steps:

1. Partition the data into training set and testing set.
2. Train a classifier (e.g. Logistic Regression, Decision Trees) using the training data.
3. Tune the model by optimizing its parameters.
4. Pass the reserved testing data set to the final model and **evaluate the model performance**.

The above steps can be executed that are available in the free, object-oriented, programming language Python https://www.python.org.

Steps to Developing a Model

In order to develop a classifier that will identify transactions as either fraudulent or normal we perform the following steps:

1. Partition the data into training set and testing set.
2. Train a classifier (e.g. Logistic Regression, Decision Trees) using the training data.
3. Tune the model by optimizing its parameters.
4. Pass the reserved testing data set to the final model and **evaluate the model performance.**

The above steps can be executed that are available in the free, object-oriented, programming language Python https://www.python.org.

Steps to Developing a Model

In order to develop a classifier that will identify transactions as either fraudulent or normal we perform the following steps:

1. Partition the data into training set and testing set.
2. Train a classifier (e.g. Logistic Regression, Decision Trees) using the training data.
3. Tune the model by optimizing its parameters.
4. Pass the reserved testing data set to the final model and **evaluate the model performance**.

The above steps can be executed that are available in the free, object-oriented, programming language Python https://www.python.org.

Steps to Developing a Model

In order to develop a classifier that will identify transactions as either fraudulent or normal we perform the following steps:

1. Partition the data into training set and testing set.
2. Train a classifier (e.g. Logistic Regression, Decision Trees) using the training data.
3. Tune the model by optimizing its parameters.
4. Pass the reserved testing data set to the final model and **evaluate the model performance**.

The above steps can be executed that are available in the free, object-oriented, programming language Python https://www.python.org.

Steps to Developing a Model

In order to develop a classifier that will identify transactions as either fraudulent or normal we perform the following steps:

1. Partition the data into training set and testing set.
2. Train a classifier (e.g. Logistic Regression, Decision Trees) using the training data.
3. Tune the model by optimizing its parameters.
4. Pass the reserved testing data set to the final model and **evaluate the model performance**.

The above steps can be executed that are available in the free, object-oriented, programming language Python https://www.python.org.

Evaluating Model Performance

There are several standard metrics available to measure the performance of a ML algorithm, however some are more appropriate than others depending on the goal.

Of the $284,807$ recorded transactions we reserve 25% for future testing. That leaves us with $213,605$ transactions to train the model on and $71,202$ transactions to test how well the model performs.

Assume that the testing set consists of $71,073$ **normal transactions** and 129 transactions labelled as **fraudulent**.

Evaluating Model Performance

There are several standard metrics available to measure the performance of a ML algorithm, however some are more appropriate than others depending on the goal.

Of the $284,807$ recorded transactions we reserve 25% for future testing. That leaves us with $213,605$ transactions to train the model on and $71,202$ transactions to test how well the model performs.

Assume that the testing set consists of $71,073$ **normal transactions** and 129 transactions labelled as **fraudulent**.

Evaluating Model Performance

There are several standard metrics available to measure the performance of a ML algorithm, however some are more appropriate than others depending on the goal.

Of the $284,807$ recorded transactions we reserve 25% for future testing. That leaves us with $213,605$ transactions to train the model on and $71,202$ transactions to test how well the model performs.

Assume that the testing set consists of $71,073$ **normal transactions** and 129 transactions labelled as **fraudulent**.

TP, TN, FP, FN

Every prediction that the model makes will be in one of the four categories:

- True Positive (TP): The transaction was fraudulent and the model correctly predicted the transaction was fraudulent.
- True Negative (TN): The transaction was normal and the model correctly predicted the transaction was normal.
- False Positive (FP): The transaction was normal but the model incorrectly predicted that the transaction was fraudulent.
- False Negative (FN): The transaction was fraudulent but the model incorrectly predicted that the transaction was normal.

All this information can be represented in a confusion matrix.

TP, TN, FP, FN

Every prediction that the model makes will be in one of the four categories:

- True Positive (TP): The transaction was fraudulent and the model correctly predicted the transaction was fraudulent.
- True Negative (TN): The transaction was normal and the model correctly predicted the transaction was normal.
- False Positive (FP): The transaction was normal but the model incorrectly predicted that the transaction was fraudulent.
- False Negative (FN): The transaction was fraudulent but the model incorrectly predicted that the transaction was normal.

All this information can be represented in a confusion matrix.

TP, TN, FP, FN

Every prediction that the model makes will be in one of the four categories:

- True Positive (TP): The transaction was fraudulent and the model correctly predicted the transaction was fraudulent.
- True Negative (TN): The transaction was normal and the model correctly predicted the transaction was normal.
- False Positive (FP): The transaction was normal but the model incorrectly predicted that the transaction was fraudulent.
- False Negative (FN): The transaction was fraudulent but the model incorrectly predicted that the transaction was normal.

All this information can be represented in a confusion matrix.

TP, TN, FP, FN

Every prediction that the model makes will be in one of the four categories:

- True Positive (TP): The transaction was fraudulent and the model correctly predicted the transaction was fraudulent.
- True Negative (TN): The transaction was normal and the model correctly predicted the transaction was normal.
- False Positive (FP): The transaction was normal but the model incorrectly predicted that the transaction was fraudulent.
- False Negative (FN): The transaction was fraudulent but the model incorrectly predicted that the transaction was normal.

All this information can be represented in a confusion matrix.

TP, TN, FP, FN

Every prediction that the model makes will be in one of the four categories:

- True Positive (TP): The transaction was fraudulent and the model correctly predicted the transaction was fraudulent.
- True Negative (TN): The transaction was normal and the model correctly predicted the transaction was normal.
- False Positive (FP): The transaction was normal but the model incorrectly predicted that the transaction was fraudulent.
- False Negative (FN): The transaction was fraudulent but the model incorrectly predicted that the transaction was normal.

All this information can be represented in a confusion matrix.

Confusion Matrix

Table: Confusion matrix

	Predicted "fraudulent"	**Predicted "normal"**
"fraudulent"	TP	FN
"normal"	FP	TN

Suppose the confusion matrix for our credit card fraud detection model is:

Table: Confusion matrix

	Predicted "fraudulent"	**Predicted "normal"**
"fraudulent"	72	57
"normal"	4	71,069

Accuracy

Accuracy measures how often the classifier makes the correct prediction.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{72 + 71,069}{72 + 71,069 + 4 + 57} = 0.99914$$

Does that mean our model is phenomenal?

NO!!!

We can see that despite having an accuracy of over 99.9% our model predicted 57 fraud cases (nearly 44%) incorrectly as normal!

This usually happens when data set is **unbalanced**.

Accuracy

Accuracy measures how often the classifier makes the correct prediction.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{72 + 71,069}{72 + 71,069 + 4 + 57} = 0.99914$$

Does that mean our model is phenomenal?

NO!!!

We can see that despite having an accuracy of over 99.9% our model predicted 57 fraud cases (nearly 44%) incorrectly as normal!

This usually happens when data set is **unbalanced**.

Accuracy

Accuracy measures how often the classifier makes the correct prediction.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{72 + 71,069}{72 + 71,069 + 4 + 57} = 0.99914$$

Does that mean our model is phenomenal?

NO!!!

We can see that despite having an accuracy of over 99.9% our model predicted 57 fraud cases (nearly 44%) incorrectly as normal!

This usually happens when data set is **unbalanced**.

Accuracy

Accuracy measures how often the classifier makes the correct prediction.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{72 + 71,069}{72 + 71,069 + 4 + 57} = 0.99914$$

Does that mean our model is phenomenal?

NO!!!

We can see that despite having an accuracy of over 99.9% our model predicted 57 fraud cases (nearly 44%) incorrectly as normal!

This usually happens when data set is **unbalanced**.

Accuracy

Accuracy measures how often the classifier makes the correct prediction.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{72 + 71,069}{72 + 71,069 + 4 + 57} = 0.99914$$

Does that mean our model is phenomenal?

NO!!!

We can see that despite having an accuracy of over 99.9% our model predicted 57 fraud cases (nearly 44%) incorrectly as normal!

This usually happens when data set is **unbalanced**.

Precision

Precision attempts to answer the question: What proportion of positive identifications was correct?

i.e., out of all the transactions predicted to be fraudulent how many of them were actually fraudulent?

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{72}{72 + 4} = 0.94736$$

Precision

Precision attempts to answer the question: What proportion of positive identifications was correct?

i.e., out of all the transactions predicted to be fraudulent how many of them were actually fraudulent?

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{72}{72 + 4} = 0.94736$$

Precision

Precision attempts to answer the question: What proportion of positive identifications was correct?

i.e., out of all the transactions predicted to be fraudulent how many of them were actually fraudulent?

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{72}{72 + 4} = 0.94736$$

Recall

Recall attempts to answer the question: What proportion of actual positives was correctly identified?

i.e., out of all transactions that were actually fraudulent, how many of them were correctly identified as fraudulent?

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{72}{72 + 57} = 0.55813$$

Recall

Recall attempts to answer the question: What proportion of actual positives was correctly identified?

i.e., out of all transactions that were actually fraudulent, how many of them were correctly identified as fraudulent?

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{72}{72 + 57} = 0.55813$$

Precision Versus Recall

As the data expert, you will have to decide which of these scores (precision or recall) is more important to optimize for your classification problem.

In this problem it would be more harmful to the financial company if the algorithm incorrectly predicts a transaction as normal when in reality it is fraudulent compared to the trade-off of having a transaction flagged as fraudulent when it is actually normal. Thus, the credit card fraud detection model is a **high recall model**.

Precision Versus Recall

As the data expert, you will have to decide which of these scores (precision or recall) is more important to optimize for your classification problem.

In this problem it would be more harmful to the financial company if the algorithm incorrectly predicts a transaction as normal when in reality it is fraudulent compared to the trade-off of having a transaction flagged as fraudulent when it is actually normal. Thus, the credit card fraud detection model is a **high recall model**.

More Metrics

Other metrics that can be used to evaluate model performance are:

- F_β Score
- AUC Score

Classification Problems
Email Spam Filter

Further Topics for Exploration

You should familiarize yourself with the following terms:

- Bias Versus Variance
- Model Complexity Graph
- Learning Curve
- k-fold Cross Validation
- Grid Search

Classification Problems
Student College Commitment Decisions

The Admission Process

The goal of the research is to develop a mathematical model that can make a prediction regarding each student's college commitment decision by classifying the student into one of two categories: **accepts admission offer** and **rejects admission offer**.

Features (Input Variables) and Output Variable

Table: List of Variables

Binary	Categorical	Numerical
Gross Commit Indicator	Application Term	GPA
Net Commit Indicator	Ethnic Background	HS Class Rank
Final Decision	Permanent State/Region	HS Class Size
Financial Aid Intent	Permanent Zip Code/Postal Code	ACT Composite Score
Scholarship	Permanent Country	SAT I Critical Reading
Legacy	Current/Most Recent School Geomarket	SAT I Math
Direct Legacy	First Source Date	SAT I Writing
First Generation Indicator	First Source Summary	SAT I Superscore
Campus Visit Indicator	Top Academic Interest	SATR EBRW
Interview	Extracurricular Interests	SATR Math
Recruited Athlete Indicator	Level of Financial Need	SATR Total
Sex		Reader Academic Rating

The Data Distribution

Table: Counts and Percentages of First-Year Commit Data

Application Year	Class of 2018	Class of 2019	Class of 2020	Class of 2021	Total
Applicants	6,071	5,911	6,409	6,775	25,166
Admits	2,549	2,659	2,948	2,845	11,001
Admit Percentage	42%	45%	46%	42%	44%
Commits	547	518	502	571	2,138
Commit Percentage	21%	19%	17%	20%	19%
Uncommits	2,002	2,141	2,446	2,274	8,863
Uncommit Percentage	79%	81%	83%	80%	81%

Acknowledgments

I would like to thank Prof. Buckmire for inviting me to Math 396 and for the opportunity to deliver this lecture.

The End