Teaching a Computer to Diagnose Cancer

An Introduction to Machine Learning

Glenn R. Fisher

April 18, 2016

IBM Mobile Innovation Lab

What is Machine Learning?

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

- **Supervised Learning**:

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

- **Supervised Learning**:
  - Data is organized into input-output pairs.

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

- **Supervised Learning**:
    - Data is organized into input-output pairs.
    - The goal is to learn the relationship between inputs and outputs.

IBM Mobile Innovation Lab

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

- **Supervised Learning**:
  - Data is organized into input-output pairs.
  - The goal is to learn the relationship between inputs and outputs.
- **Unsupervised Learning**:

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

- **Supervised Learning**:
  - Data is organized into input-output pairs.
  - The goal is to learn the relationship between inputs and outputs.
- **Unsupervised Learning**:
  - Data is *not* organized into input-output pairs.

IBM Mobile Innovation Lab

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

- **Supervised Learning**:
  - Data is organized into input-output pairs.
  - The goal is to learn the relationship between inputs and outputs.
- **Unsupervised Learning**:
  - Data is *not* organized into input-output pairs.
  - The goal is to learn the relationship between data points.

IBM Mobile Innovation Lab

What is Machine Learning?

Machine learning uses computer programs to learn relationships in data.

- **Supervised Learning**:
  - Data is organized into input-output pairs.
  - The goal is to learn the relationship between inputs and outputs.
- **Unsupervised Learning**:
  - Data is *not* organized into input-output pairs.
  - The goal is to learn the relationship between data points.
  - This is also known as data mining, clustering, etc.

Supervised Learning:

Supervised Learning:

- Data: $(\vec{x_1}, y_1), (\vec{x_2}, y_2), ..., (\vec{x_n}, y_n)$

Supervised Learning:

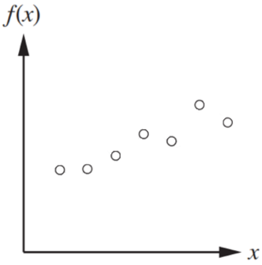
- Data: $(\vec{x_1}, y_1), (\vec{x_2}, y_2), ..., (\vec{x_n}, y_n)$
- Assumption: The data was generated by some function, $f(\vec{x_i}) = y_i$

Supervised Learning:

- Data: $(\vec{x_1}, y_1), (\vec{x_2}, y_2), ..., (\vec{x_n}, y_n)$
- Assumption: The data was generated by some function, $f(\vec{x_i}) = y_i$
- Goal: Learn the function, $f(\vec{x_i})$

Supervised Learning:

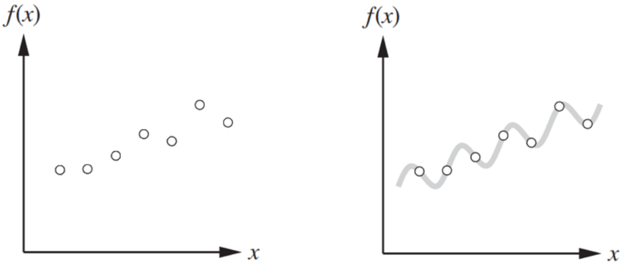
- Data: $(\vec{x_1}, y_1), (\vec{x_2}, y_2), \ldots, (\vec{x_n}, y_n)$
- Assumption: The data was generated by some function, $f(\vec{x_i}) = y_i$
- Goal: Learn the function, $f(\vec{x_i})$



IBM Mobile Innovation Lab

Supervised Learning:

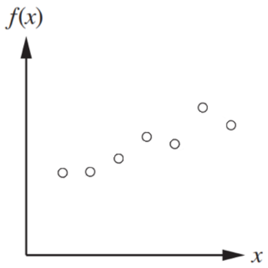
- Data: $(\vec{x_1}, y_1), (\vec{x_2}, y_2), \ldots, (\vec{x_n}, y_n)$
- Assumption: The data was generated by some function, $f(\vec{x_i}) = y_i$
- Goal: Learn the function, $f(\vec{x_i})$



IBM Mobile Innovation Lab

Supervised Learning:

- Data: $(\vec{x_1}, y_1), (\vec{x_2}, y_2), ..., (\vec{x_n}, y_n)$
- Assumption: The data was generated by some function, $f(\vec{x_i}) = y_i$
- Goal: Learn the function, $f(\vec{x_i})$

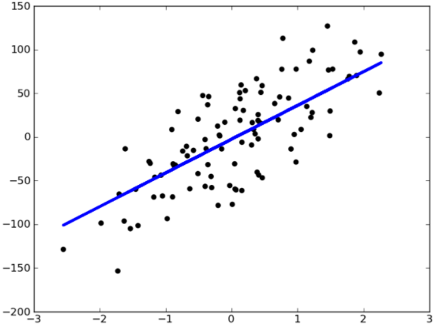


IBM Mobile Innovation Lab

Types of Supervised Learning

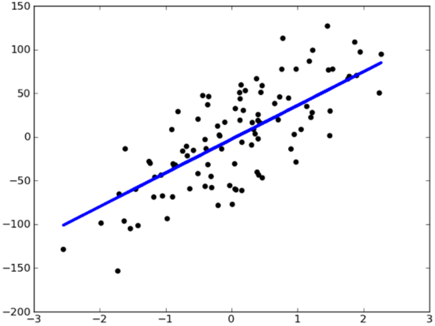
Types of Supervised Learning

Regression: Numerical Outputs

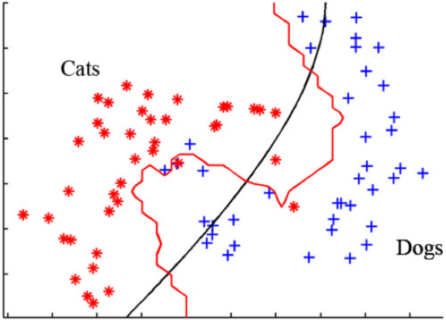


IBM Mobile Innovation Lab

Regression: Numerical Outputs

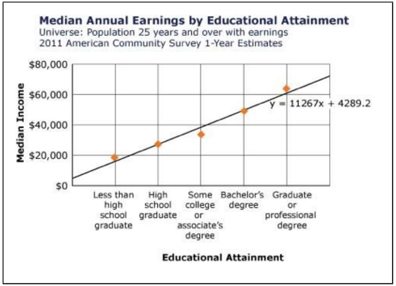


Classification: Categorical Outputs

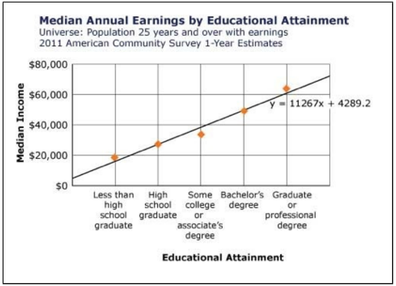


Supervised Learning Examples

Supervised Learning Examples



Input: Years of Education
Output: Annual Income
Application: Predict Income

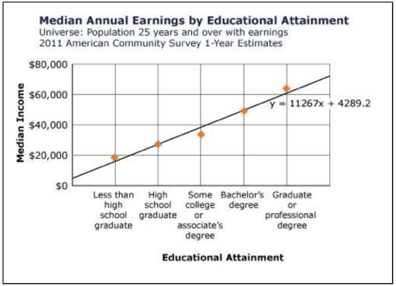


Input: Years of Education
Output: Annual Income
Application: Predict Income

Input: Pictures of Written Digits
Output: Intended Digit
Application: Zip-Code Mail Sorting

IBM Mobile Innovation Lab

Supervised Learning Examples



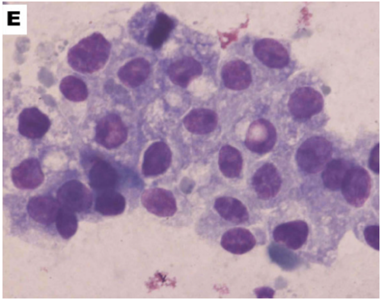
Input: Years of Education
Output: Annual Income
Application: Predict Income

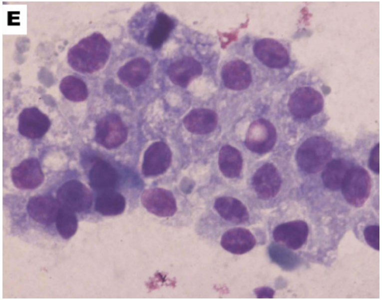
Input: Pictures of Written Digits
Output: Intended Digit
Application: Zip-Code Mail Sorting

Input: Environment Sensors
Output: Driving Instructions
Application: Self-Driving Cars

IBM Mobile Innovation Lab

Identifying Cancer Cells





Are these cancer cells?

1. Yes, they are malignant.
2. No, they are benign.

IBM Mobile Innovation Lab

Identifying Cancer Cells: Problem Formulation

- What are the inputs?

- What are the inputs? $\Rightarrow$ The shape of each cell.

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs?

Identifying Cancer Cells: Problem Formulation

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).
- What are our assumptions?

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).
- What are our assumptions? $\Rightarrow$ There is a relationship between cell shape and cancer.

Identifying Cancer Cells: Problem Formulation

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).
- What are our assumptions? $\Rightarrow$ There is a relationship between cell shape and cancer.
- What is our goal?

IBM Mobile Innovation Lab

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).
- What are our assumptions? $\Rightarrow$ There is a relationship between cell shape and cancer.
- What is our goal? $\Rightarrow$ Learn the relationship between cell shape and cancer.

Identifying Cancer Cells: Problem Formulation

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).
- What are our assumptions? $\Rightarrow$ There is a relationship between cell shape and cancer.
- What is our goal? $\Rightarrow$ Learn the relationship between cell shape and cancer.
- What data will we use to solve our goal?

IBM Mobile Innovation Lab

Identifying Cancer Cells: Problem Formulation

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).
- What are our assumptions? $\Rightarrow$ There is a relationship between cell shape and cancer.
- What is our goal? $\Rightarrow$ Learn the relationship between cell shape and cancer.
- What data will we use to solve our goal? $\Rightarrow$ Wisconsin Diagnostic Breast Cancer (WDBC).

IBM Mobile Innovation Lab

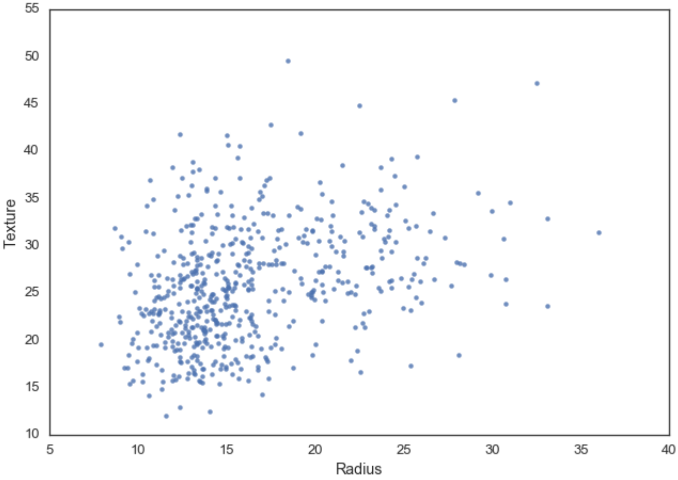
Identifying Cancer Cells: Problem Formulation

- What are the inputs? $\Rightarrow$ The shape of each cell.
- What are the outputs? $\Rightarrow$ A diagnosis (malignant or benign).
- What are our assumptions? $\Rightarrow$ There is a relationship between cell shape and cancer.
- What is our goal? $\Rightarrow$ Learn the relationship between cell shape and cancer.
- What data will we use to solve our goal? $\Rightarrow$ Wisconsin Diagnostic Breast Cancer (WDBC).

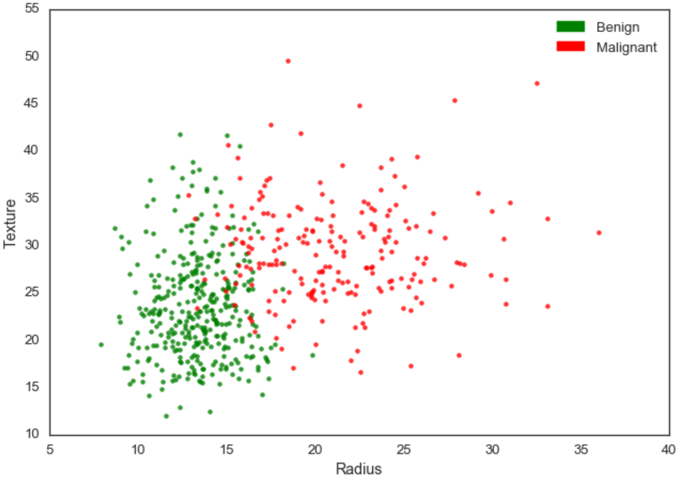
Diagnosis	Radius	Texture
Malignant	16.08	21.82
Malignant	25.28	25.59
Benign	14.48	21.82
...	...	...
Malignant	20.92	34.69

Identifying Cancer Cells: Data Exploration

Identifying Cancer Cells: Data Exploration

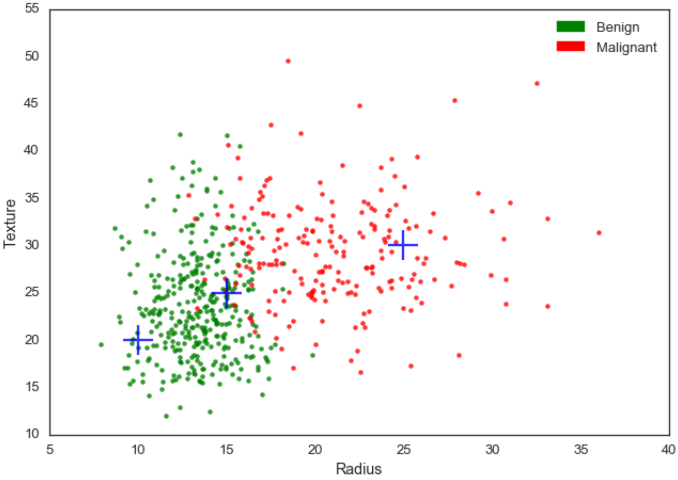


Identifying Cancer Cells: Data Exploration



IBM Mobile Innovation Lab

Identifying Cancer Cells: Data Exploration

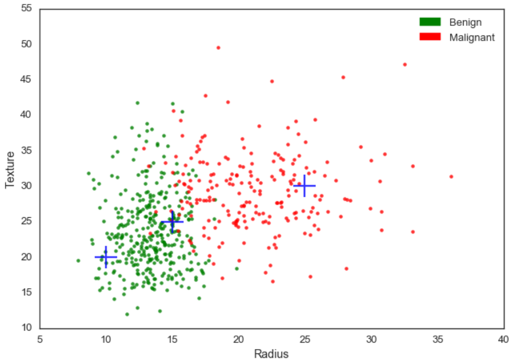


Identifying Cancer Cells: k-Nearest Neighbors Algorithm

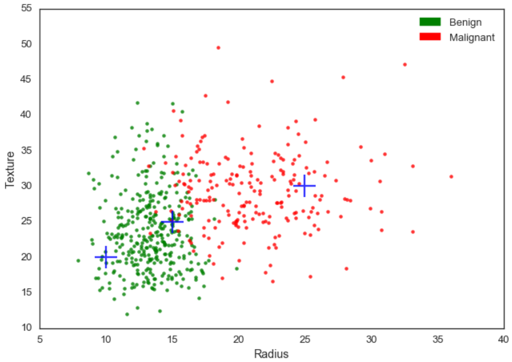
Identifying Cancer Cells: k-Nearest Neighbors Algorithm

k-Nearest Neighbors Algorithm: Classify an observation according to the class of its neighbors.

k-Nearest Neighbors Algorithm: Classify an observation according to the class of its neighbors.

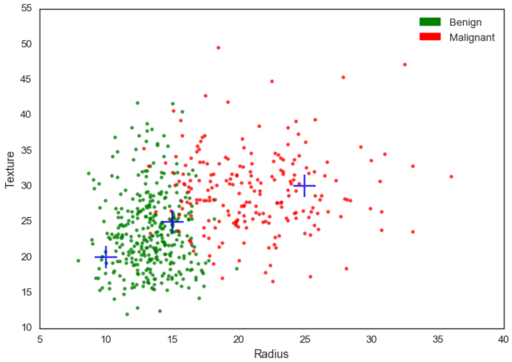


k-Nearest Neighbors Algorithm: Classify an observation according to the class of its neighbors.



How It Works:

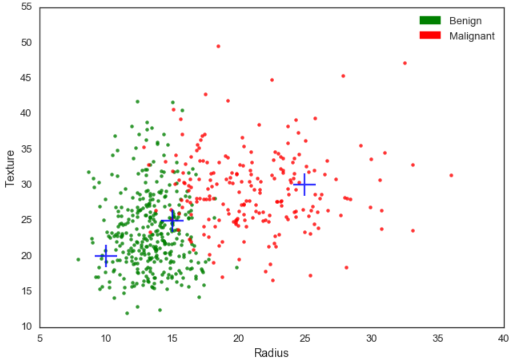
k-Nearest Neighbors Algorithm: Classify an observation according to the class of its neighbors.



How It Works:

1. Consider observations as points in space.

k-Nearest Neighbors Algorithm: Classify an observation according to the class of its neighbors.



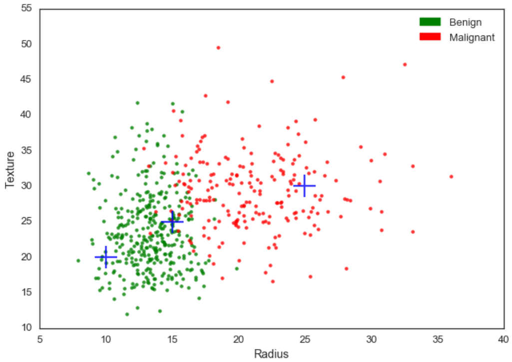
How It Works:

1. Consider observations as points in space.
2. Given a new observation to classify, find its k nearest neighbors.

IBM Mobile Innovation Lab

Identifying Cancer Cells: k-Nearest Neighbors Algorithm

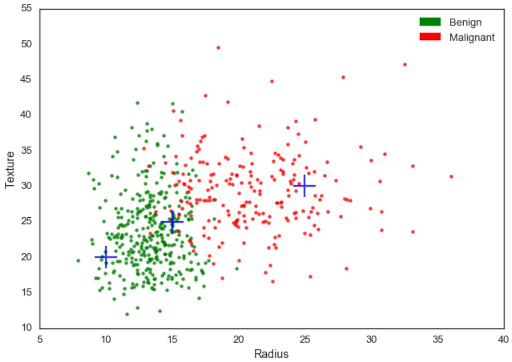
k-Nearest Neighbors Algorithm: Classify an observation according to the class of its neighbors.



How It Works:

1. Consider observations as points in space.
2. Given a new observation to classify, find its k nearest neighbors.
3. Among these neighbors, find the most common class.

k-Nearest Neighbors Algorithm: Classify an observation according to the class of its neighbors.



How It Works:

1. Consider observations as points in space.
2. Given a new observation to classify, find its k nearest neighbors.
3. Among these neighbors, find the most common class.
4. Use that class to classify the new observation.

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Training

Identifying Cancer Cells: Model Training

1. Load the dataset.

IBM Mobile Innovation Lab

1. Load the dataset.

```
columns_to_features = { 1: "Diagnosis",
                        22: "Radius",
                        23: "Texture"}
```

1. Load the dataset.

```
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}

features_to_keep = columns_to_features.values()
```

1. Load the dataset.

```
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}

features_to_keep = columns_to_features.values()

wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")
```

Identifying Cancer Cells: Model Training

1. Load the dataset.

```
columns_to_features = { 1: "Diagnosis",
                        22: "Radius",
                        23: "Texture"}

features_to_keep = columns_to_features.values()

wdbc = pd.read_csv("wdbc.data", header = None) \
        .rename(columns = columns_to_features) \
        .filter(features_to_keep, axis = 1) \
        .replace("M", "Malignant") \
        .replace("B", "Benign")
```

2. Split dataset into separate training and test sets.

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Training

1. Load the dataset.

```
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}

features_to_keep = columns_to_features.values()

wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")
```

2. Split dataset into separate training and test sets.

```
x_full = wdbc.drop("Diagnosis", axis = 1)
y_full = wdbc["Diagnosis"]
x_train, x_test, y_train, y_test = sklearn.cross_validation.train_test_split(
    x_full, y_full, test_size = 0.3, random_state = 3)
```

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Training

1. Load the dataset.

```
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}

features_to_keep = columns_to_features.values()

wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")
```

2. Split dataset into separate training and test sets.

```
x_full = wdbc.drop("Diagnosis", axis = 1)
y_full = wdbc["Diagnosis"]
x_train, x_test, y_train, y_test = sklearn.cross_validation.train_test_split(
    x_full, y_full, test_size = 0.3, random_state = 3)
```

3. Train a k-nearest neighbors classifier.

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Training

1. Load the dataset.

```
columns_to_features = { 1: "Diagnosis",
                        22: "Radius",
                        23: "Texture"}

features_to_keep = columns_to_features.values()

wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")
```

2. Split dataset into separate training and test sets.

```
x_full = wdbc.drop("Diagnosis", axis = 1)
y_full = wdbc["Diagnosis"]
x_train, x_test, y_train, y_test = sklearn.cross_validation.train_test_split(
    x_full, y_full, test_size = 0.3, random_state = 3)
```

3. Train a k-nearest neighbors classifier.

```
model = sklearn.neighbors.KNeighborsClassifier().fit(x_train, y_train)
```

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Evaluation

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.
2. Compare each prediction with that observation's actual diagnosis.

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.
2. Compare each prediction with that observation's actual diagnosis.
3. Compute model accuracy as the fraction of correct predictions.

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.
2. Compare each prediction with that observation's actual diagnosis.
3. Compute model accuracy as the fraction of correct predictions.

This is a straightforward computation in Python:

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.
2. Compare each prediction with that observation's actual diagnosis.
3. Compute model accuracy as the fraction of correct predictions.

This is a straightforward computation in Python:

```
predictions = model.predict(x_test)
```

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.
2. Compare each prediction with that observation's actual diagnosis.
3. Compute model accuracy as the fraction of correct predictions.

This is a straightforward computation in Python:

```
predictions = model.predict(x_test)

sklearn.metrics.accuracy_score(predictions, y_test)
```

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.
2. Compare each prediction with that observation's actual diagnosis.
3. Compute model accuracy as the fraction of correct predictions.

This is a straightforward computation in Python:

```
predictions = model.predict(x_test)

sklearn.metrics.accuracy_score(predictions, y_test)
```

We find that our simple k-nearest neighbors classifier has 95% accuracy!

IBM Mobile Innovation Lab

Identifying Cancer Cells: Model Evaluation

We want to know how well our model generalizes. How accurately can it predict a diagnosis?

To answer this question, we will:

1. Use the trained model to predict a diagnosis for each observation in the test set.
2. Compare each prediction with that observation's actual diagnosis.
3. Compute model accuracy as the fraction of correct predictions.

This is a straightforward computation in Python:

```
predictions = model.predict(x_test)
sklearn.metrics.accuracy_score(predictions, y_test)
```

We find that our simple k-nearest neighbors classifier has 95% accuracy!

We just trained a computer to correctly diagnose breast cancer cells 95% of the time.

IBM Mobile Innovation Lab

Identifying Cancer Cells: The Program

Identifying Cancer Cells: The Program

```
# load dataset
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}
features_to_keep = columns_to_features.values()
wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")
```

IBM Mobile Innovation Lab

Identifying Cancer Cells: The Program

```
# load dataset
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}
features_to_keep = columns_to_features.values()
wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")

# split dataset into training and test sets
x_full = wdbc.drop("Diagnosis", axis = 1)
y_full = wdbc["Diagnosis"]
x_train, x_test, y_train, y_test = sklearn.cross_validation.train_test_split(
    x_full, y_full, test_size = 0.3, random_state = 3)
```

IBM Mobile Innovation Lab

```
# load dataset
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}
features_to_keep = columns_to_features.values()
wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")

# split dataset into training and test sets
x_full = wdbc.drop("Diagnosis", axis = 1)
y_full = wdbc["Diagnosis"]
x_train, x_test, y_train, y_test = sklearn.cross_validation.train_test_split(
    x_full, y_full, test_size = 0.3, random_state = 3)

# train a k-nearest neighbors model
model = sklearn.neighbors.KNeighborsClassifier().fit(x_train, y_train)
```

Identifying Cancer Cells: The Program

```
# load dataset
columns_to_features = { 1: "Diagnosis",
                       22: "Radius",
                       23: "Texture"}
features_to_keep = columns_to_features.values()
wdbc = pd.read_csv("wdbc.data", header = None) \
         .rename(columns = columns_to_features) \
         .filter(features_to_keep, axis = 1) \
         .replace("M", "Malignant") \
         .replace("B", "Benign")

# split dataset into training and test sets
x_full = wdbc.drop("Diagnosis", axis = 1)
y_full = wdbc["Diagnosis"]
x_train, x_test, y_train, y_test = sklearn.cross_validation.train_test_split(
    x_full, y_full, test_size = 0.3, random_state = 3)

# train a k-nearest neighbors model
model = sklearn.neighbors.KNeighborsClassifier().fit(x_train, y_train)

# evaluate the model's accuracy
predictions = model.predict(x_test)
sklearn.metrics.accuracy_score(predictions, y_test)
```

IBM Mobile Innovation Lab

Recap

Let's review some of the big ideas:

Recap

Let's review some of the big ideas:

- Machine Learning: Use computer programs to learn relationships in data.

Recap

Let's review some of the big ideas:

- Machine Learning: Use computer programs to learn relationships in data.
- Supervised Learning: Learn relationships between inputs and outputs.

IBM Mobile Innovation Lab

Recap

Let's review some of the big ideas:

- Machine Learning: Use computer programs to learn relationships in data.
- Supervised Learning: Learn relationships between inputs and outputs.
  - Regression: Numerical outputs.

IBM Mobile Innovation Lab

Recap

Let's review some of the big ideas:

- Machine Learning: Use computer programs to learn relationships in data.
- Supervised Learning: Learn relationships between inputs and outputs.
  - Regression: Numerical outputs.
  - Classification: Categorical outputs.

IBM Mobile Innovation Lab

Recap

Let's review some of the big ideas:

- Machine Learning: Use computer programs to learn relationships in data.
- Supervised Learning: Learn relationships between inputs and outputs.
  - Regression: Numerical outputs.
  - Classification: Categorical outputs.
- Unsupervised Learning: Learn relationships between data points.

IBM Mobile Innovation Lab

Recap

Let's review some of the big ideas:

- Machine Learning: Use computer programs to learn relationships in data.
- Supervised Learning: Learn relationships between inputs and outputs.
  - Regression: Numerical outputs.
  - Classification: Categorical outputs.
- Unsupervised Learning: Learn relationships between data points.
- k-Nearest Neighbors Algorithm: Classify an observation according to its neighbors.

Recap

Let's review some of the big ideas:

- Machine Learning: Use computer programs to learn relationships in data.
- Supervised Learning: Learn relationships between inputs and outputs.
  - Regression: Numerical outputs.
  - Classification: Categorical outputs.
- Unsupervised Learning: Learn relationships between data points.
- k-Nearest Neighbors Algorithm: Classify an observation according to its neighbors.
- Diagnosing Cancer: We trained a model to diagnose cancer with 95% accuracy.

IBM Mobile Innovation Lab