ISA 414 – Managing Big Data

Lecture 13 – Data Analysis

Supervised Learning (Part II)

Dr. Arthur Carvalho

arthur.carvalho@miamioh.edu

MIAMI UNIVERSITY

Announcements

- **Information Systems & Analytics Seminar Series**
  - **Applying Machine Learning & Artificial Intelligence to Ecommerce/Digital Business Operations across Marketing, Merchandising, and Customer Support**
    - Tuesday, October 12, 2021, 7:00 pm
    - https://miamioh.zoom.us/j/81146372341?pwd=SzJIdk1PTWdhVDhBcXZOSTZFVVV3UT09
  - By **Matt Fritz**
    - Senior Director, Machine Learning & Artificial Intelligence @ Samsung Electronics

Announcements

- Python cheat sheet available on Canvas

- Assignment 3
  - I messed up with the server and connection string last Thursday
  - Please redownload the assignment description

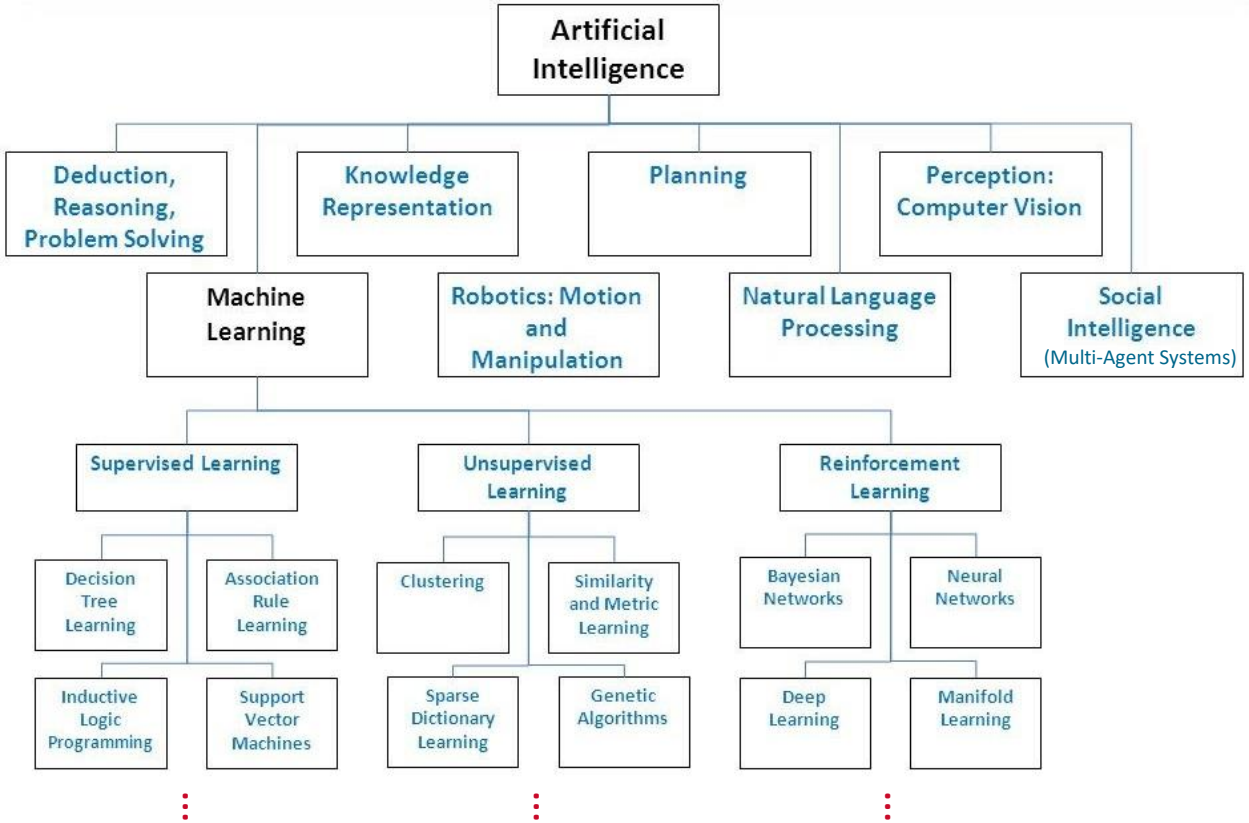
Lecture Objectives

- Learn about predictive models for regression problems

Lecture Instructions

- Download the following files available on Canvas
  - *Lecture 13.ipynb*
  - *energy_data.csv*
  - *Description of the variables.xlsx*
- Place the above files inside the same folder
- Open the file *Lecture 13.ipynb* with VS Code

AI


Artificial Intelligence
Deduction, Reasoning, Problem Solving
Knowledge Representation
Planning
Perception: Computer Vision
Machine Learning
Robotics: Motion and Manipulation
Natural Language Processing
Social Intelligence
(Multi-Agent Systems)
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Decision Tree Learning
Association Rule Learning
Clustering
Similarity and Metric Learning
Bayesian Networks
Neural Networks
Inductive Logic Programming
Support Vector Machines
Sparse Dictionary Learning
Genetic Algorithms
Deep Learning
Manifold Learning

Machine Learning

- The solutions to the category of problems we will be working on are based on a paradigm called supervised learning
  - Assumption: there is a clear target variable whose values we are trying to predict

Problem	Supervised	Unsupervised
Classification	X	
Regression	X	
Clustering		X
Co-occurrence grouping		X
Profiling		X
…	…	…

Previous class → Classification

Our focus today → Regression

Supervised Learning

- Classification (or class probability estimation)
  - Goal: determine which class a new observation (likely) belongs to
    - The target variable is qualitative (categorical/factor)
  - Example: given a data set containing data about previous and current clients, can we build a statistical model to suggest an electricity tariff to a new client?
    - Categorical target

Supervised Learning

- Regression problems
  - Goal: predict the value of a target variable using a model structure that relates the target with informative attributes
    - The target variable is quantitative (numeric)
  - Example: what will be the annual electricity consumption for a 4-bedroom house in Oxford having 2 adults and 2 kids, an electric vehicle, …?
    - Clear numeric target (consumption)

Supervised Learning

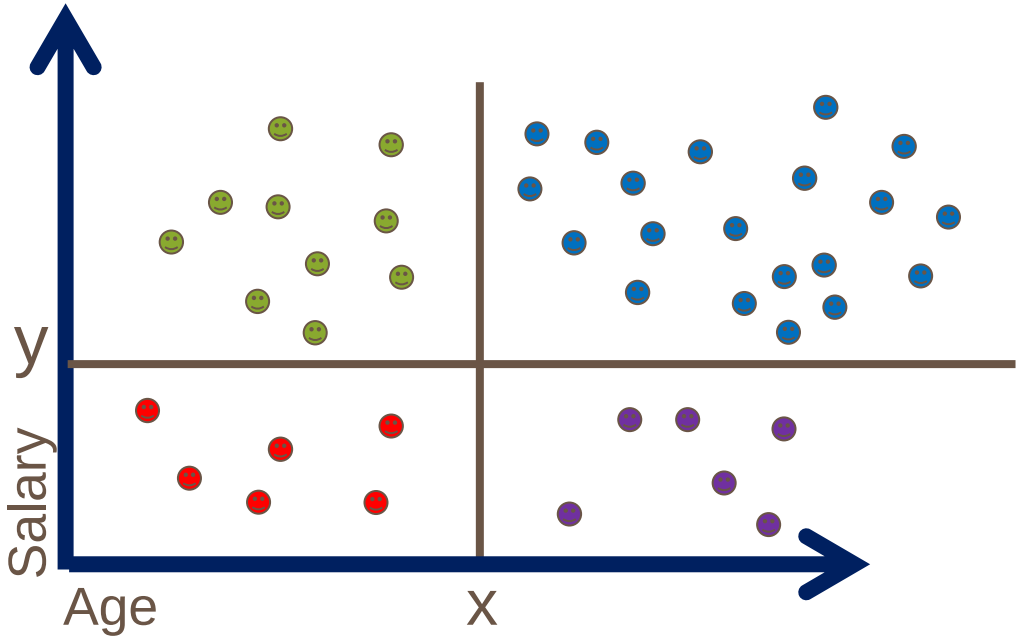
- Illustrative problem
  - An energy (electricity) provider has historical data with information about current clients
    - *E.g.*, Age, salary, marital status, *etc.*
  - Each client (observation) in the data set is associated with an annual-consumption value
  - Business question:
    - Can we build a model to predict the annual consumptions of future and current clients?
      - Why? To determine whether new power plants are required

Regression Problems

- Potential solution:
  - Translate this problem into a regression problem
  - Collect the data
    - Query internal databases
  - Build a statistical model to predict future consumption
  - Use the model to predict the consumption for all new and current clients

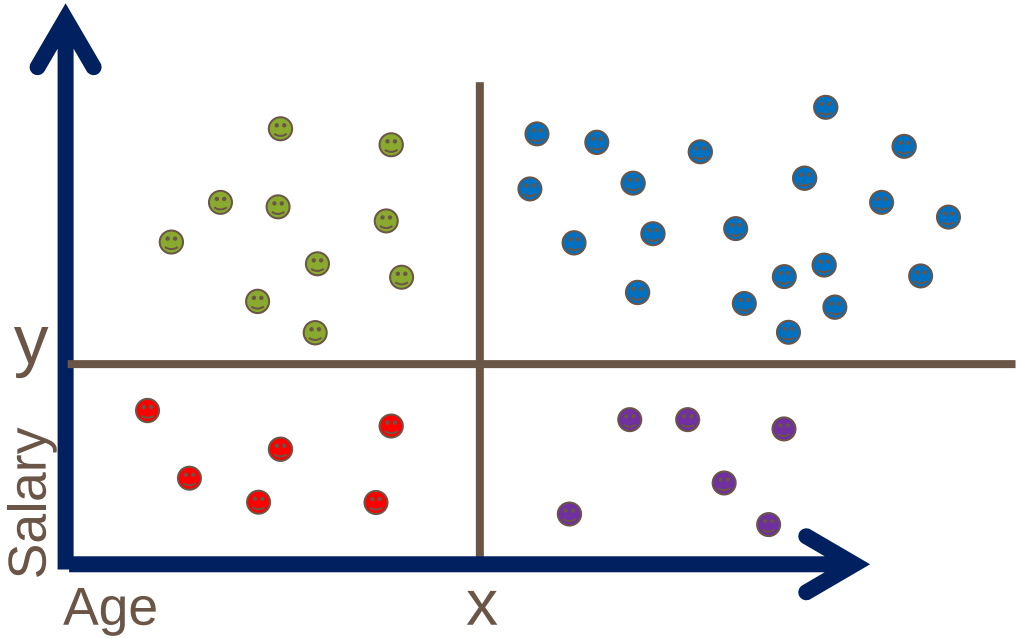
Regression Problems

- Decision trees and random forests can also be applied to regression problems
  - For example, a predicted value can simply be the average consumption for all the clients in a partition created by a decision tree



Regression Problems

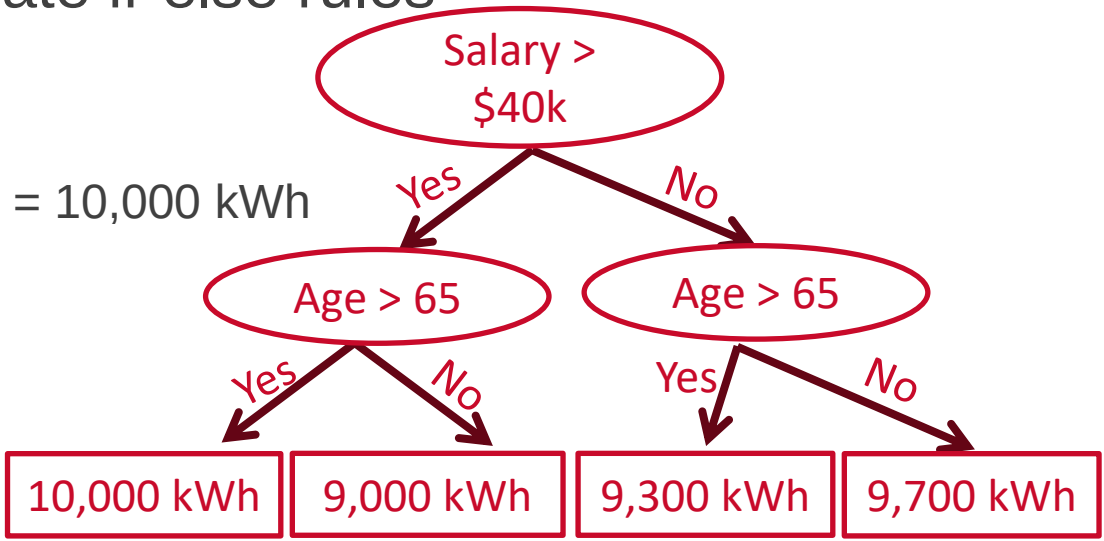
- Example:
  - New client's age > x AND new client's salary > y
    - Prediction = average consumption of the blue clients
  - New client's age > x AND new client's salary < y
    - Prediction = average consumption of the purple clients
  - New client's age < x AND new client's salary > y
    - Prediction = average consumption of the green clients
  - New client's age < x AND new client's salary < y
    - Prediction = average consumption of the red clients



Regression Problems

- The resulting model is often called *regression tree, i.e.*, a decision tree for regression problems
- Regression trees also create if-else rules

E.g., IF 'Salary > $40K' AND
'Age > 65'
THEN PredictedConsumption = 10,000 kWh



Regression Problems

- There are many algorithms for building regression trees from a data set
  - These are part of a field called *machine learning*
    - Beyond the scope of this course

- We take a more applied approach
  - Let's build a regression tree to predict annual consumption

Regression Trees

- Regression Trees in Python
  - Let's build a regression tree using the module sklearn
  - Load pandas and the data set we use today

```
import pandas as pd
energy_data = pd.read_csv("energy_data.csv")
```

- **Before analyzing the data, always check whether each attribute is of the right type (class)**
  - If it is not, then change attribute types

```
energy_data.dtypes
```

Regression Trees

- Let's create dummies for the qualitative predictors
  - Remember that we do this because sklearn does not accept qualitative predictors, regardless of the model

```
energy_data = pd.get_dummies(energy_data,
                             columns = ["MaritalStatus", "IncomeLevel", "DwellingArea",
                                        "HasChildren", "SolarRoof", "ShiftableLoad",
                                        "AttitudeSustainability", "Tariff" ],
                             drop_first = True)
```

  - Note that our target is now *AnnualConsumption*
    - Hence, we create dummies for *Tariff*

Regression Trees

- Let's split the data now intro training and test sets

```
from sklearn.model_selection import train_test_split

x = energy_data.drop(columns=["AnnualConsumption"])
y = energy_data["AnnualConsumption"]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.34)
```

Regression Trees

- Building a decision tree using the training set

```
from sklearn.tree import DecisionTreeRegressor

tree_model = DecisionTreeRegressor()

tree_model = tree_model.fit(x_train, y_train)
```

  - Pay attention to the function DecisionTreeRegressor instead of DecisionTreeClassifier

Regression Trees

- Let's now use our model to predict the consumption of the clients left out in the test set

```
y_pred = tree_model.predict(x_test)
```

- Now, we have two lists of interest
  - y_pred contains the predictions made by our model
  - y_test contains the true consumption values
  - How do we compare them?
    - Can we simply check how often these values agree with other?

Model Evaluation

- Example: Consider the values in the table
  - What is the accuracy of the model if we just compare whether the predicted values are equal to the true value?
  - We must use a **distance metric** to measure how accurate models are in regression problems
    - Different metrics, *e.g.,* MSE, RMSE, MAD, …

Predictions on the test data $(\hat{y}_t)$	True values in the test data (y_t)
10,009 kWh	9,993 kWh
11,057 kWh	10,990 kWh
12,781 kWh	13,163 kWh
11,800 kWh	11,789 kWh
9,783 kWh	9,999 kWh

Model Evaluation

- Approach #1: Mean Squared Error (MSE)

$$MSE = \frac{1}{n} \times \sum_{t} (y_t - \hat{y}_t)^2$$

  - MSE penalizes large errors because the errors are squared
  - It is not in the same units as the data

	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Client 7	Client 8	Client 9	Client 10
Consumption in 1,000 kWh	2.444	2.801	2.993	2.963	3.046	3.033	2.637	2.319	2.287	2.38
Forecast	**2.3565**	**2.399**	**2.6225**	**2.897**	**2.978**	**3.0045**	**3.0395**	**2.835**	**2.478**	**2.303**
Error	**0.0875**	**0.402**	**0.3705**	**0.066**	**0.068**	**0.0285**	**-0.4025**	**-0.516**	**-0.191**	**0.077**
Error^2	**0.0077**	**0.1616**	**0.1373**	**0.0044**	**0.0046**	**0.0008**	**0.1620**	**0.2663**	**0.0365**	**0.0059**

MSE ≈ (0.0077 + 0.1616 + 0.1373 + 0.0044 + 0.0046 + 0.0008 + 0.1620 + 0.2663+ 0.0365 + 0.0059)/10
≈ 0.07871

Model Evaluation

- Approach #2: Root Mean Square Error (RMSE)

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \times \sum_{t} (y_t - \hat{y}_t)^2}$$

  - RMSE is in the same units as the data
  - It assigns more weight to large deviations such as outliers (same as MSE)
    - Large squared differences become larger
    - Small squared differences become smaller

	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Client 7	Client 8	Client 9	Client 10
Consumption in 1,000 kWh	2.444	2.801	2.993	2.963	3.046	3.033	2.637	2.319	2.287	2.38
Forecast	**2.3565**	**2.399**	**2.6225**	**2.897**	**2.978**	**3.0045**	**3.0395**	**2.835**	**2.478**	**2.303**
Error	**0.0875**	**0.402**	**0.3705**	**0.066**	**0.068**	**0.0285**	**-0.4025**	**-0.516**	**-0.191**	**0.077**
Error^2	**0.0077**	**0.1616**	**0.1373**	**0.0044**	**0.0046**	**0.0008**	**0.1620**	**0.2663**	**0.0365**	**0.0059**

$$RMSE \approx \sqrt{0.07871} \approx 0.28$$

Model Evaluation

- Approach #3: Mean Absolute Deviation (MAD) or Mean Absolute Error (MAE)

$$MAD = \frac{1}{n} \times \sum_{t} |y_t - \hat{y}_t|$$

  - MAD is in the same units as the data and it is easy to interpret
  - Assigns less weight to outliers

	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Client 7	Client 8	Client 9	Client 10
Consumption in 1,000 kWh	2.444	2.801	2.993	2.963	3.046	3.033	2.637	2.319	2.287	2.38
Forecast	**2.3565**	**2.399**	**2.6225**	**2.897**	**2.978**	**3.0045**	**3.0395**	**2.835**	**2.478**	**2.303**
Error	**0.0875**	**0.402**	**0.3705**	**0.066**	**0.068**	**0.0285**	**-0.4025**	**-0.516**	**-0.191**	**0.077**
|Error|	**0.0875**	**0.402**	**0.3705**	**0.066**	**0.068**	**0.0285**	**0.4025**	**0.516**	**0.191**	**0.077**

MAD ≈ (0.0875 + 0.402 + 0.3705 + 0.066 + 0.068 + 0.0285 + 0.4025 + 0.5160 + 0.191 + 0.077)/10
≈ 0.22

Model Evaluation

- Approach #4: Mean Absolute Percentage Error (MAPE)

$$MAPE = 100 \times \frac{1}{n} \times \sum \frac{|y_t - \hat{y}_t|}{|y_t|}$$

- MAPE compares the absolute errors to the magnitude of the estimated quantity
- It is easy to interpret because it is a percentage, independent of the unit of the target variable

	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Client 7	Client 8	Client 9	Client 10
Consumption in 1,000 kWh	2.444	2.801	2.993	2.963	3.046	3.033	2.637	2.319	2.287	2.38
Forecast	**2.3565**	**2.399**	**2.6225**	**2.897**	**2.978**	**3.0045**	**3.0395**	**2.835**	**2.478**	**2.303**
Error	**0.0875**	**0.402**	**0.3705**	**0.066**	**0.068**	**0.0285**	**-0.4025**	**-0.516**	**-0.191**	**0.077**
$\lvert Error \rvert / \lvert y_t \rvert$	**0.0358**	**0.1435**	**0.1238**	**0.0223**	**0.0223**	**0.0094**	**0.1526**	**0.2225**	**0.0835**	**0.0324**

MAPE ≈ (0.0358 + 0.1435 + 0.1238 + 0.0223 + 0.0223 + 0.0094 + 0.1526 + 0.2225 + 0.0835 + 0.0324)*100/10
≈ 8.48%

Model Evaluation

- Which error measure should one use?
  - It depends on a few factors
    - Penalize outliers, interpretability, *etc*.
  - It is common practice to report forecast errors using more than one measure when evaluating models
    - For example, RMSE plus MAD

Model Evaluation

- Let's evaluate our model using the MAE metric
  - mean_absolute_error function from the metrics submodule

```
from sklearn.metrics import mean_absolute_error
mean_absolute_error(y_test, y_pred)
```

- Interpretation
  - A *MAE* equal to, say, 887 means that the predictions from the model are expected to be 887 units away from the true predictions
    - *I.e.*, our model is on average 887 kWh off

Random Forests

- Let's build a second model using the same data
  - Random forests: 200 trees

```
from sklearn.ensemble import RandomForestRegressor

model = RandomForestRegressor(n_estimators=200)

model = model.fit(x_train,y_train)
```

  - Pay attention to the function RandomForestRegressor instead of RandomForestClassifier

Random Forests

- Let's build a second model using the same data
  - Predictions and evaluation

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

#evaluation
mean_absolute_error(y_test, y_pred)
```

Model Evaluation

- Recall that we are evaluating models based only on accuracy
  - Interpretation is another possible evaluation aspect
  - Predictions by regression/decision trees are explainable
    - For example, look at the function plot_tree in the sklearn.tree module
    - It might take a long time to run that function

Summary

- Summary
  - Data-analytics problems: regression problems
  - Regression trees and random forests
  - Evaluation: training and test sets
- Useful references
  - http://www.r2d3.us/visual-intro-to-machine-learning-part-1/
  - https://docs.google.com/presentation/d/1kSuQyW5DTnkVaZEjGYCkfOxvzCqGEFzWBy4e9Uedd9k/edit
- Next class: text mining