Subject:	CSCI
Course Number:	9
Descriptive Title:	Practical Data Science
Division:	Mathematical Sciences
Department:	Computer Science
Course Disciplines:	Computer Science
Catalog Description:	This course examines computation, inferential thinking, and the issues of utilizing real -world data to solve challenging data science problems. This intermediate-level class bridges between Foundations of Data Science and the methods currently u sed in industry. Students will understand the data science life-cycle and learn many of the principles and techniques of data science, such as spannin g algorithms, statistics, machine learning, visualization, and data systems.
Prerequisite:	CSCI-8 and CSCI-14 or CSCI-1
Co-requisite:	
Recommended Preparation:	
Enrollment Limitation:	
Hours Lecture (per week):	3
Hours Laboratory (per week):	3
Outside Study Hours:	6
Total Course Hours:	108
Course Units:	4
Grading Method:	Letter Grade only
Credit Status:	Credit, degree applicable
Transfer CSU:	Yes
Effective Date:	Fall 2023
Transfer UC:	Yes
Effective Date:	Pending
General Education ECC:	
Term:	
Other:	
CSU GE:	
Term:	
Other:	
IGETC:	
Term:	

Other:	
Student Learning Outcomes:	**SLO #1: Using Specifications** Students will write correct computer code that will produce appropriate visualizations and analysis of data. **SLO #2: Tracing the Execution** Students, when given a code segment, will be able to trace the execution and give the output. **SLO #3: Identifying and Correcting Errors** Students, when given a data science code with errors, will be able to identify and corr ect the problems. **SLO #4: Explaining Concepts** Students will be able to explain data science concepts related to causality versus corre lation, hypothesis testing, prediction and classification.
Course Objectives:	1. Students will explain and justify conclusions about data and draw robust concl usions based on incomplete information. 2. Students will develop written computer code, demonstrating computational t hinking and skills, for visualizing and analyzing data. 3. Students will make and justify predictions based on machine learning. 4. Students will interpret and communicate data and results using a vast array of real-world examples.
Major Topics:	**I. Data Science Lifecyle (4 hours, lecture)** A. Data Scope B. Data Design C. Modeling and Estimation **II. Tabular Data (4 hours, lecture)** A. Subsetting B. Aggregating C. Joining D. Transforming **III. Structured Query Language (6 hours, lecture)**

A. Creating Tables
B. Querying Rows

C. Querying Groups

D. Relationship types
E. Joining

F. Subsampling

IV. Data Cleaning (5 hours, lecture)

A. Modifying DataFrames
B. Cleaning messy data

C. Exploratory Data Analysis

V. Hypothesis Testing (3 hours, lecture)

A. Comparing Distributions

B. Decisions and Uncertainty

C. A/B Testing

VI. Data Granularity (4 hours, lecture)

A. Aggregations

B. Combining Data

C. Permutation Tests

VII. Missing Data (4 hours, lecture)

A. Identifying Missing Data

B. Handling Missing Data

C. Single-Valued Imputation

D. Probabilistic Imputation

VIII. Data Collection (9 hours, lecture)

A. Using existing data

B. HTTP Requestions

C. Parsing HTML

D. Regular Expressions

E. Natural Language Processing

IX. Feature engineering (6 hours, lecture)

A. Data Pipelines

B. Scikit-Learn

C. Prediction

X. Models (3 hours, lecture)

A. Statistical Models

B. Modeling pipelines

XI. Bias and Variance (3 hours, lecture)

A. Model Inference Quality

B. Model Prediction Quality

C. Cross Validation

D. Parameter Search

XII. Evaluating Models (3 hours, lecture)

A. Evaluation metrics

B. Parity Measures

C. Fairness in Machine Learning

XIII. Data Science Lifecyle (4 hours, lab)

A. Data Scope
B. Data Design

C. Modeling and Estimation

XIV. Tabular Data (4 hours, lab)

A. Subsetting
B. Aggregating

C. Joining

D. Transforming

XV. Structured Query Language (6 hours, lab)

A. Creating Tables
B. Querying Rows

C. Querying Groups

D. Relationship types
E. Joining

F. Subsampling

XVI. Data Cleaning (5 hours, lab)

A. Modifying DataFrames
B. Cleaning messy data

C. Exploratory Data Analysis

XVII. Hypothesis Testing (3 hours, lab)

A. Comparing Distributions

B. Decisions and Uncertainty

C. A/B Testing

XVIII. Data Granularity (4 hours, lab)

A. Aggregations

B. Combining Data

C. Permutation Tests

XIX. Missing Data (4 hours, lab)

A. Identifying Missing Data

B. Handling Missing Data

C. Single-Valued Imputation

D. Probabilistic Imputation

XX. Data Collection (9 hours, lab)

A. Using existing data

B. HTTP Requestions

	C. Parsing HTML D. Regular Expressions E. Natural Language Processing **XXI. Feature engineering (6 hours, lab)** A. Data Pipelines B. Scikit-Learn C. Prediction **XXII. Models (3 hours, lab)** A. Statistical Models B. Modeling pipelines **XXIII. Bias and Variance (3 hours, lab)** A. Model Inference Quality B. Model Prediction Quality C. Cross Validation D. Parameter Search **XXIV. Evaluating Models (3 hours, lab)** A. Evaluation metrics B. Parity Measures C. Fairness in Machine Learning
Total Lecture Hours:	54
Total Laboratory Hours:	54
Total Hours:	108
Primary Method of Evaluation:	2) Problem solving demonstrations (computational or non-computational)
Typical Assignment Using Primary Method of Evaluation:	Create the following functions using numpy methods satisfying the requirements given in each part. Your solutions should **not** contain any loops or list comprehensions. • A function arr_1 that takes in a numpy array and adds to each element the square-root of the index of each element. • A function arr_2 that takes in a numpy array of integers and returns a boolean array (i. e. an array of booleans) whose ith element is True if and only if the ith element of the input array is a perfect square.

	• A function arr_3 that takes in a numpy array of stock prices per share on succ essive days in USD and returns an array of growth rates. That is, the ith numb er of the output array should contain the rate of growth in stock price betwee n the day to the day. The growth rate should be a proportion, rounded to the nearest hundredth. • Suppose: o A is a numpy array of stock prices per share for a company on successi ve days in USD o You start each day with \$20 to buy as much stock as possible on that day. o Any money left- over after a given day is saved for possibly buying stock on a future da y. o Create a function arr_4 that takes in A and returns the day on which y ou can buy at least one share from 'left- over' money. If this never happens, return - 1. The first stock purchase occurs on day 0. *Note: you cannot buy fract ions of a share of stock* . o *Example:* If the stock price is \$3 every day, then the answer is 'day 1': ▪ day 0: buy six stocks with \$20, \\$2 are added to the leftover , and your total leftover is currently \$2, so you can't buy one extra share ▪ day 1: buy six stocks with \$20, another \\$2 are added to the leftover, and your total leftover is now \$4, so you can now b uy one extra share, return day1. Hint: np. cumsum may be helpful for this question
Critical Thinking Assignment 1:	**Messy Data** The file in data/messy. txt contains personal information from a fictional website that a user scraped from webserver logs. Within this dataset, there are four fields that inte rest you: 1. Email Addresses (assume they are alphanumeric user-names and domain-names), 2. Social Security Numbers 3. Bitcoin Addresses (alpha-numeric strings of long length) 4. Street Addresses Create a function extract_personal that takes in a string like open('data/messy. txt'). r ead() and returns a tuple of four separate lists containing values of the 4 pieces of info rmation listed above (in the order given). Do **not** keep empty values. *Hint* : There are multiple "delimiters" in use in the file; there are few enough of them t hat you can safely determine what they are. *Note:* Since this data is messy/corrupted, your function will be allowed to miss ~5% of the records in each list. Good spot checking using certain useful substrings (e. g.@ for emails) should help assure correctness! Your function will be tested on a sample of th e file messy. txt.
Critical Thinking Assignment 2:	**Modeling**

	In this question, you will train two different prediction models on Galton's child-height dataset from lecture and explore different ways in how overfitting can appear. **Part 1: Decision Tree Regressor** • A decision tree regressor is trained similar to decision tree classifiers: the splits of the tree are created by minimizing the variance of the target values of the (training) data in the leaves given by making the split in question. • A decision tree regressor predicts the target value of a (new) observation based on the average target value of the training observations lying in the same leaf node. • One parameter of a decision tree regressor that affects model complexity is the *depth* of the tree. We will explore this parameter in this question. • Create a function tree_reg_perf that takes in a dataframe like galton and outputs a dataframe where each row contains the *RMSE* of a trained decision tree regressor on the training set and test set, indexed by the depth of the decision tree (depth=1,2,3,...,20). (i.e. you should train 20 different decision trees with varying depths and put the train/test error into a dataframe). *Note* (Optional question good for studying): How is this overfitting and why? What type of variance is causing it? What is the best choice of depth? Plot the dataframe above to help answer these questions. **Part 2: k-Nearest Neighbor Regressor** • A k-NN Regressor predicts the target value of a (new) observation by computing the average value of the k-closest observations in the training set. • One parameter of a k-NN regressor that affects model performance is the number of neighbors averaged over. We will explore this parameter in this question. • Create a function knn_reg_perf that takes in a dataframe like galton and outputs a dataframe where each row contains the *RMSE* of a trained k-NN regressor on the training set and test set, indexed by the number of neighbors (k=1,2,3,...,20). *Note* (Optional question good for studying): How is this overfitting and why? What type of variance is causing it? What is the best choice for the number of neighbors? Plot the dataframe above to help answer these questions.
Other Evaluation Methods:	Homework Problems, Objective Exam, Term or Other Papers, Written Homework
If Other:	Programming assignments
Instructional Methods:	Lab, Lecture
If other:	
Work Outside of Class:	Problem solving activity, Required reading, Study, Written work (such as essay/composition/report/analysis/research)
If Other:	
Up-To-Date Representative Textbooks:	An Introduction to Statistical Learning , Gareth James; Daniela Witten; Trevor Hastie; Robert Tibshirani, Springer Science, New York, 2 nd edition, 2021 Practical Data Science , Aaron Fraenkel, from https://afraenkel.github.io/practical-data-science/

Alternative Textbooks:	
Required Supplementary Readings:	
Other Required Materials:	
Requisite	Prerequisite
Category	sequential
Requisite course:	CSCI-8 and CSCI-14 or CSCI-1
Requisite and Matching skill(s): Bold the requisite skill. List the corresponding course objective under each skill(s).	**Develop algorithms using a programming language, use one-dimensional arrays within a programming language, develop subprograms using a p rogramming language** CSCI 1 - Design programming solutions requiring the storage and manipulation of large amounts of data (with random access ability during execution cycle), using single and multi-dimensional arrays, such as numerical, string, char, and structure types. CSCI 14 - Design programs utilizing lists, dictionaries, and strings. **Visualization and analyzing data; making and justifying prediction** CSCI 8 - Develop written computer code, demonstrating computational thinking and skills, for visualizing and analyzing data. CSCI 8 - Make and justify predictions based on machine learning.
Requisite Skill:	
Requisite Skill and Matching skill(s): Bold the requisite skill(s). if applicable	
Requisite course:	
Requisite and Matching skill(s): Bold the requisite skill. List the corresponding course objective under each skill(s).	
Requisite Skill:	
Requisite Skill and Matching skill(s): Bold the requisite skill. List the corresponding	

course objective under each skill(s). if applicable	
Enrollment Limitations and Category:	
Enrollment Limitations Impact:	
Course Created by:	Solomon Russell
Date:	11/19/2021
Board Approved:	8/15/2022