# Homework: Week 2

Tic-tac-toe (or noughts and crosses) is a simple strategy game in which two players take turns placing a mark on a 3x3 board, attempting to make a row, column, or diagonal of three with their mark. In this homework, we will use the tools we've covered in the past two weeks to create a tic-tac-toe simulator and evaluate basic winning strategies.

In the following exercises, we will learn to create a tic-tac-toe board, place markers on the board, evaluate if either player has won, and use this to simulate two basic strategies.

Click the link to download the [Jupyter Notebook for the Week 2 Homework](https://courses.edx.org/assets/courseware/v1/8df1f9ae7d887b8f72782cf588000c51/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/HW2.ipynb).

**Homework: Case Study 1**

A cipher is a secret code for a language. In this case study, we will explore a cipher that is reported by contemporary Greek historians to have been used by Julius Caesar to send secret messages to generals during times of war.

The Caesar cipher shifts each letter of a message to another letter in the alphabet located a fixed distance from the original letter. If our encryption key were 1, we would shift **h** to the next letter **i**, **i** to the next letter **j**, and so on. If we reach the end of the alphabet, which for us is the space character, we simply loop back to **a**. To decode the message, we make a similar shift, except we move the same number of steps backwards in the alphabet.

In the five exercises of this homework, we will create our own Caesar cipher, as well as a message decoder for this cipher.

# Homework: Case Study 2

In the six exercises of this case study, we will find and plot the distribution of word frequencies for different translations of Hamlet. Perhaps the distribution of word frequencies of Hamlet depends on the translation -- let's find out!

For this case study, the functions count\_words\_fast and word\_stats are defined as in the Case 2 Videos (Videos 3.2.1 through 3.2.6). The code for these functions, which you will need for the following exercises, is given here:

import os

import pandas as pd

import numpy as np

from collections import Counter

def count\_words\_fast(text):

text = text.lower()

skips = [".", ",", ";", ":", "'", '"', "\n", "!", "?", "(", ")"]

for ch in skips:

text = text.replace(ch, "")

word\_counts = Counter(text.split(" "))

return word\_counts

def word\_stats(word\_counts):

num\_unique = len(word\_counts)

counts = word\_counts.values()

return (num\_unique, counts)

Click the link to download the [Jupyter Notebook for Case Study 2 External link](https://courses.edx.org/assets/courseware/v1/e32dce434a05259f142a3f53f937d971/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/HW3-2.ipynb).

# Homework: Case Study 3

In the nine exercises in this case study, we will analyze a dataset consisting of an assortment of wines classified as "high quality" and "low quality" and will use -Nearest Neighbors classification to determine whether or not other information about the wine helps us correctly guess whether a new wine will be of high quality.

You will need this sample code for the case study:

import numpy as np, random, scipy.stats as ss

def majority\_vote\_fast(votes):  
    mode, count = ss.mstats.mode(votes)  
    return mode

def distance(p1, p2):  
    return np.sqrt(np.sum(np.power(p2 - p1, 2)))

def find\_nearest\_neighbors(p, points, k=5):  
    distances = np.zeros(points.shape[0])  
    for i in range(len(distances)):  
        distances[i] = distance(p, points[i])  
    ind = np.argsort(distances)  
    return ind[:k]

def knn\_predict(p, points, outcomes, k=5):  
    ind = find\_nearest\_neighbors(p, points, k)  
    return majority\_vote\_fast(outcomes[ind])[0]

Click the link to download the [Jupyter Notebook for Case Study 3](https://courses.edx.org/assets/courseware/v1/e9f4e53be88e15dfcd9d2e5be97e30e0/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/HW3-3.ipynb).

# Homework: Case Study 4

In this case study, we have prepared step-by-step instructions for you on how to prepare plots in Bokeh, a library designed for simple, interactive plotting.  We will demonstrate Bokeh by continuing the analysis of Scotch whiskies.

Before you start the exercises, run the following code to get everything you'll need set up:

from sklearn.cluster.bicluster import SpectralCoclustering

import numpy as np, pandas as pd

whisky = pd.read\_csv("https://courses.edx.org/asset-v1:HarvardX+PH526x+2T2019+type@asset+block@whiskies.csv", index\_col=0)

correlations = pd.DataFrame.corr(whisky.iloc[:,2:14].transpose())

Click the link to download the [Jupyter Notebook for Case Study 4 External link](https://courses.edx.org/assets/courseware/v1/9f429adeb86fc04f35106cce151adb65/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/HW4-1.ipynb). The whisky dataset can be downloaded from [within the courseware at this link External link](https://courses.edx.org/assets/courseware/v1/ea5ce720d534302a26749ac02704c54a/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/whiskies.csv) and [from anywhere at this link](https://courses.edx.org/asset-v1:HarvardX+PH526x+2T2019+type@asset+block@HW4-1.ipynb).

# Homework: Case Study 5

In this case study, we will continue taking a look at patterns of flight for each of the three birds in our dataset. We will group the flight patterns by bird and date, and plot the mean altitude for these groupings.

In order to do the homework, you should start by reading in the data using the following code:

import pandas as pd

import numpy as np

birddata = pd.read\_csv("https://courses.edx.org/asset-v1:HarvardX+PH526x+2T2019+type@asset+block@bird\_tracking.csv", index\_col=0)

birddata.head()

You can download the [Jupyter Notebook for Case Study 5 at this link External link](https://courses.edx.org/assets/courseware/v1/44f5c73939cc585e1cba23e67946646f/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/HW4-2.ipynb). The bird tracking data can be found at [this link within the courseware External link](https://courses.edx.org/assets/courseware/v1/6184eb0f87c7b58db1a5c336e436ed09/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/bird_tracking.csv) and [at this link from anywhere](https://courses.edx.org/asset-v1:HarvardX+PH526x+2T2019+type@asset+block@HW4-2.ipynb).

# Homework: Case Study 7, Part 1

The [movie dataset on which this case study is based](https://www.kaggle.com/tmdb/tmdb-movie-metadata) is a database of 5000 movies catalogued by [The Movie Database (TMDb)](https://www.themoviedb.org/?language=en). The information available about each movie is its budget, revenue, rating, actors and actresses, etc. In this case study, we will use this dataset to determine whether any information about a movie can predict the total revenue of a movie. We will also attempt to predict whether a movie's revenue will exceed its budget.

In Part 1 of this case study, we will inspect, clean, and transform the data.

You can download the [Jupyter Notebook for Case Study 7, Part 1 at this link](https://courses.edx.org/assets/courseware/v1/920dda65408300ffc5ea2b41440109c3/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/HW5-1.ipynb).

# Homework: Case Study 7, Part 2

The [movie dataset on which this case study is based](https://www.kaggle.com/tmdb/tmdb-movie-metadata) is a database of 5000 movies catalogued by [The Movie Database (TMDb)](https://www.themoviedb.org/?language=en). The information available about each movie is its budget, revenue, rating, actors and actresses, etc. In this case study, we will use this dataset to determine whether any information about a movie can predict the total revenue of a movie. We will also attempt to predict whether a movie's revenue will exceed its budget.

In Part 2, we will use the dataset prepared in Part 1 for an applied analysis. We will primarily use the two models we discussed in the Week 5 videos: linear and logistic regression and random forests to perform prediction and classification. We will use these methods to predict revenue, and we will use logistic regression to classify whether a movie was profitable.

This code will get your environment set up for Case Study 7, Part 2:

# DO NOT EDIT THIS CODE  
import pandas as pd  
import numpy as np

from sklearn.model\_selection import cross\_val\_score  
from sklearn.linear\_model import LinearRegression  
from sklearn.linear\_model import LogisticRegression  
from sklearn.ensemble import RandomForestRegressor  
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score  
from sklearn.metrics import r2\_score

import matplotlib.pyplot as plt

import warnings  
warnings.filterwarnings("ignore")

# EDIT THIS CODE TO LOAD THE SAVED DF FROM THE LAST HOMEWORK  
df = pd.read\_csv('movies\_clean.csv')

You can download the [Jupyter Notebook for Case Study 7, Part 2 at this link](https://courses.edx.org/assets/courseware/v1/32a19809cd0c7dd98d815f69da9dd162/asset-v1:HarvardX+PH526x+2T2021+type@asset+block/HW5-2.ipynb).