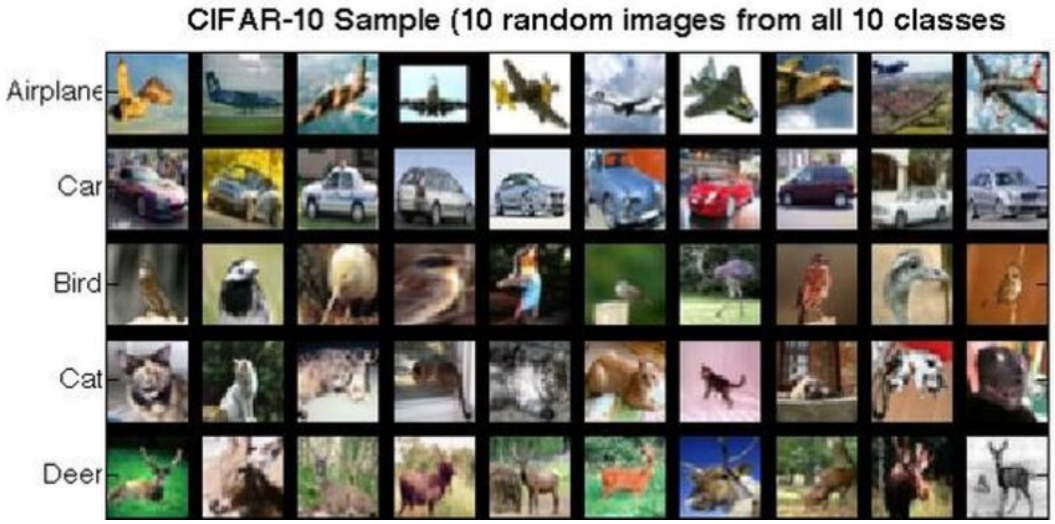
MEng in

Operations Research and Information Engineering

HW #1: Probability and Machine Learning Review



Subject: CS 5787

Professor: Alex Jaimes

Author(s): Will Daugherty-Miller **wld37**

Due date: February 3rd

CORNELL TECH

Your homework solution must be typed. Your submission must cite any references used (including articles, books, code, websites, and personal communications). All solutions must be written in your own words, and you must program the algorithms yourself. If you do work with others, you must list the people you worked with. Submit your solutions as a PDF to Canvas.

Problem 1

Recall these rules from probability: Bayes' rule is
$P(B|A) = P(A|B)P(B).P(A)$

Joint probability's relationship with conditional probability is
$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$

If two events A and B are independent then their joint probability is $P(A \cap B) = P(A)P(B)$. If two events A and B are not mutually exclusive then,

$P(A \cup B) = P(A) + P(B)P(A \cap B)$

If two events A and B are mutually exclusive then $P(AB) = 0$.
Hint: None of the questions in problem 1 have the same answer.

In this question, assume that the kangaroos have equal probability of having children that are male or female. Kangaroos almost always give birth to a single offspring.

Let $P(O = F)$ be the probability of the older offspring being female and let $P(Y = F)$ be the probability of the younger offspring being female.

1 Suppose a kangaroo has two children. What is the probability that both are female? Show your derivation.
Solution: We can begin by defining sets and subsets based on what we are interested in for the given problem statement.

- U is the set of all possible events (universe)
- A is the set of possibilities where both children are female

$$U = ((g, g), (g, b), (b, g), (b, b))$$
$$A = (g, g)$$

We can then use Bayes Rule to solve the problem:

$$P(A) = \frac{A}{U}$$
$$= \frac{1}{4}$$

2 Suppose a kangaroo has two children and the oldest is female. What is the probability that both are female? Show your derivation.
Solution: Building off from question 1, we can keep our previous definitions and define another set B.

- B is the set of possibilities that the oldest child is female

$$U = ((g,g),(g,b),(b,g),(b,b))$$
$$A = (g,g)$$
$$B = (g,g),(g,b)$$

We can then use Bayes Rule to solve the problem:

$$P(A \cap B) = \frac{\frac{A \cap B}{U}}{\frac{B}{U}}$$
$$= \frac{1}{2}$$

3 Suppose a kangaroo has two children and at least one of them is female. What is the probability that both are female? Show your derivation.

Solution:

With question 3, we can redefine B but keep the other sets the same.

- B is the set of possibilities that at least one of the children is female.

$$U = ((g,g),(g,b),(b,g),(b,b))$$
$$A = (g,g)$$
$$B = (g,g),(g,b),(b,g)$$

We can then use Bayes Rule to solve the problem:

$$P(A \cap B) = \frac{\frac{A \cap B}{U}}{\frac{B}{U}}$$
$$= \frac{1}{3}$$

Disclaimer: Work from this problem was referenced from classwork in ORIE 5530: Modeling Under Uncertainty and from Introduction to Probability Models by Sheldon Ross. Work was then consulted/compared with Marcelo Fuentes, Michelle Wang, Nick Pacia, Renata Anastasia, Alejandro Castellano and Phil Parvaneh.

Problem 2

1 Are there differences between "Machine Learning algorithm" and "classification algorithm" and if so, what are they?

Solution:

We define "classification algorithms" as a subset of "machine learning algorithms". This is to say that all "classification algorithms" are "machine learning algorithms", but not all "machine learning algorithms" are "classification algorithms". Machine learning algorithms as a whole fall into either supervised or unsupervised, or reinforcement categories. Focusing on classification, we note that this falls into the supervised category where the aim is to correctly predict categorical labels for given data.

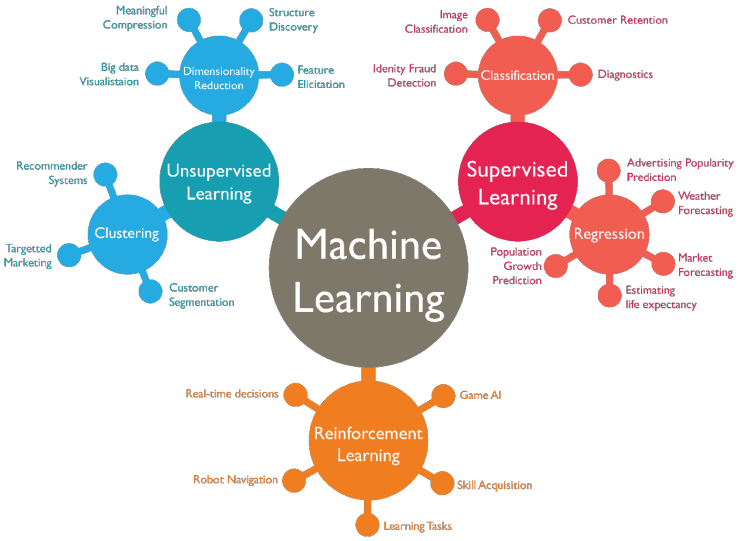


Figure 1: Overview of ML algorithmns

ML Algorithms Overview

2 Explain linear regression and logistic regression, their differences, and when you would apply each.

Solution:

We start by defining what each form of regression. Linear regression is a statistical method to to model the relationship between a dependant variable and one or more independent variables. By doing so, linear regression aims to find the best fit (in a straight line y =mx +b), through a set of data points. It is also used to predict the outcome of a continuous variable.

Logistic regression on the other hand predicts the binary outcome of independent variables. It accomplishes this by by modelling the probability that the outcome is 1 given the values of independent variables.

Now, we can turn to the differences between the two regression algorithms which can primarily be boiled down to the type of outcome variable they are used to predict(binary vs continuous variables).

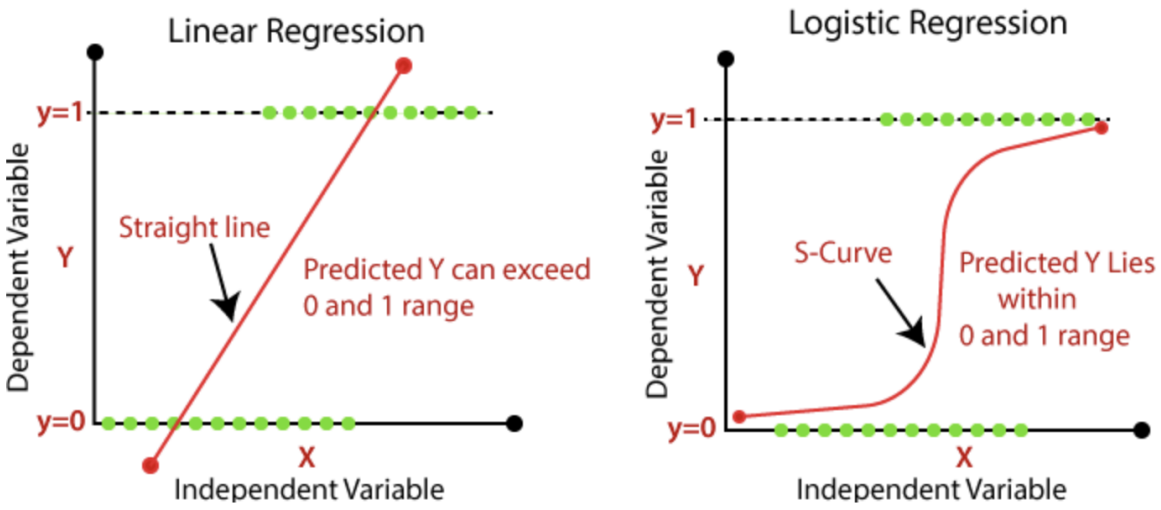


Figure 2: Comparison of linear vs logistic regression

sources: Comparison Examples of when logistic regression can be applied can be for predicting diseases in patients, defaults on loans, detecting spam emails or whether users will change hotel bookings. Logistic Regression Use Cases Conversely, linear regression to predict housing prices(like on kaggle), the amount of rainfall in a region, astronomical data analysis, calculating relationships in biological systems, or look at engine performance for cars and other vehicles.

Linear Regression Use Cases

3 Pick a regression loss function, explain how it works, and give an example. What are the advantages/disadvantages of the function you picked?

Solution:

One regression loss function is Mean Squared Error(MSE), which calculates the average squared difference between predicted values and actual values. The formula is as follows:

$$\frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

- n is the number of observations
- Y_i is the number of observed values
- $\hat{Y}_i$ is the number of predicted values

Explanation of common loss functions To use MSE, we can turn to a previous cited instance of predicting house prices. After we have used linear regression For predicting house prices, we can then evaluate our regression model with MSE (lower values being ideal). Below is a sample code for implementing MSE:

In [1]:
```
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, predictions)) print('MSE
    :', metrics.mean_squared_error(y_test, predictions)) print('RMSE:', np
    .sqrt(metrics.mean_squared_error(y_test, predictions)))
```

Out[1]:
```
OUTPUT

MAE: 82288.22251914957
MSE: 10460958907.209501
RMSE: 102278.82922291153
```

The following code was borrowed from this article just to illustrate how MSE is implemented. MSE Implementation

The advantages of MSE are that it is easy to implement and is ideal for ensuring the trained model has no outlier predictions with large errors.

The disadvantages of MSE are that it can be effected by the scale of variables and the squaring feature of the loss function can magnify bad predictions.

4 Read about Gradient Descent and explain it using a real life example (snowboarding?)
Solution:
We can think about gradient descent easily in terms of snowboarding. For snowboarding, we can think of gradient descent by finding the steepest and fastest way down the slope. After reaching the top of the mountain the boarder will start descending and make adjustments to their route as needed. The decision for the adjustments will be based on how steep the slope of the trail. There are some issues trails may have shallow or flat sections before descending and not all trails may lead to the bottom of the mountain. The goal will be accomplished when the boarder has reached the main lodge/landing area. Translating to machine learning we note that the mountain is the loss function and the boarder is the algorithm which is trying to minimize the loss. Another way to view this is a tree climber finding the fastest way down to the ground.

Figure 3: Vail ski resort

Vail trail sign Simple gradient explanation

5 You're working on solving a problem where you can use AI to make binary decisions given a set of inputs. How would you compute the errors your system might make, and how would you decide when your system is good enough to be deployed?
Solution:

To asses the state of the system we can split the data used into test and training sets respectively. Then using the training set, we can aptly train the system. After training has finished we can test the system with the test set. Once testing has completed, we can observe the system's responses and evaluate the binary classifiers performance through any one of the following metrics:

- precision
- recall
- F1-score

With any of these metrics, we can observe how the system is performing and even compare its performance to other systems.

For deciding whether my system has met the threshold for deployment, we can look at the metrics and then consider other factors such as industry, penalties for inaccuracy and cost to deploy. These factors would determine whether the proposed solution meets industry/regulatory guidelines, is cost effective and additionally ethically sound. After this a further look into test pilots and or simulations might be in order to observe any unaccounted for effects(see twitter cropping controversy) or any unaccounted for features/use cases.

Twitter Controversy

Disclaimer: Work from this problem was referenced from previous machine learning classes and the writing was improved upon using Chatgpt. Work was then consulted/compared with Marcelo Fuentes, Michelle Wang, Nick Pacia, Renata Anastasia, Alejandro Castellano and Phil Parvaneh.

Problem 3

1. **Installing Anaconda and PyTorch** Most assignments can be completed either by running and testing code on your local machine (see "Local installation") or executing the code on the cloud through Google Colab. If you have time, you might want to try both options (running code locally and on Colab) so that you gain experience and figure out which workflow suits you best.

2. **Local installation:**
   Python 3.6+, numpy, scipy, and matplotlib are necessary for the homeworks in this class. You can obtain all of these packages in one go by installing Anaconda. For your text editor, we recommend VSCode, although if you have another preference you're welcome to use any editor you'd like.
   After installing your Python 3.6+ environment along with the other toolboxes, you may optionally install PyTorch, but it will not be required until Homework 2.

3. **Google Colab:**
   Google Colab is a free hosted version of Jupyter Notebook. Unlike Python files which are executed as an entire script, notebooks are executed in blocks of code called cells. To get started, go to the Colab homepage at https://colab.research.google.com/. Create a new notebook. Type in a line of Python code and run it (and start the notebook) by pressing Shift + Enter.

4. **Visualizing CIFAR-10:**
   The CIFAR-10 dataset contains 60,000 RGB images from 10 categories. It can be found here: ref. It can alternatively be loaded via HuggingFace datasets (make sure to 'pip install datasets' before importing).Using the first CIFAR-10 training batch file, display the first three images from each of the 10 categories as a 3×10 image array. The images are stored as rows, and you will need to reshape them into $32 \times 32 \times 3$ images if you load up the raw data yourself. It is okay to use the PyTorch toolbox for loading them or you can roll your own.

   Solution: We can use a variety of approachs to solve this problem, two such solutions are to load the cifar10 dataset using Pytorch or Huggingface. By using Pytorch, we can define the following function and pull the data from torchvision.

In [2]:
```
## Load the CIFAR-10 dataset
cifar10 = datasets.CIFAR10(root='path/to/data', train=True, download=True,
     transform=transforms.ToTensor())

def pytorch_cifar():
    """Plots the cifar10 dataset from Pytorch

    Args:

    Returns:
      10x3 plot of the first three images in the cifar10 dataset
    """
    # Create a subplot of 3 rows and 10 columns
    fig, axs = plt.subplots(3, 10, figsize=(15, 6))

    # Create a dictionary to store the first three images from each
    category
    categories = {i: [] for i in range(10)}

    # Iterate through the dataset
    for data, label in cifar10:
        if len(categories[label]) < 3:
            categories[label].append(data)

    # Plot the images from the dictionary
    for i, category in enumerate(categories):
        for j, img in enumerate(categories[category]):
            ax = axs[j][i]
            ax.imshow(img.permute(1, 2, 0))
            ax.axis('off')
            ax.set_title(cifar10.classes[category])

    return plt.show() #Returns the plot
```

In [3]:
```
pytorch_cifar()
```

The result of this function can be seen below.

Figure 4: 3x 10 array for cifar10 dataset

The alternate approach for displaying the cifar10 dataset is using huggingface. (Note that the images are different possibly due to a shuffling effect).

In [4]:
```
def huggingface_cifar10():
    """Plots the cifar10 dataset from huggingface
    Args:

    Returns:
      10x3 plot of the first three images in the cifar10 dataset
    """
    dataset = load_dataset('cifar10')
    df = pd.DataFrame(dataset["train"])
    img_by_cat = []
    for i in range(10):
        img_by_cat.append(list(df.loc[df['label']==i][:3]["img"]))

    fig, axs = plt.subplots(3, 10, figsize=(15, 6))
    # Plot the images from the dictionary
    for i in range(10):
        for j in range(3):
            ax = axs[j][i]
            ax.imshow(img_by_cat[i][j])
            ax.axis('off')

    return plt.show() #Returns the plot
```

In [5]:
```
huggingface_cifar10()
```

The results can be seen below.

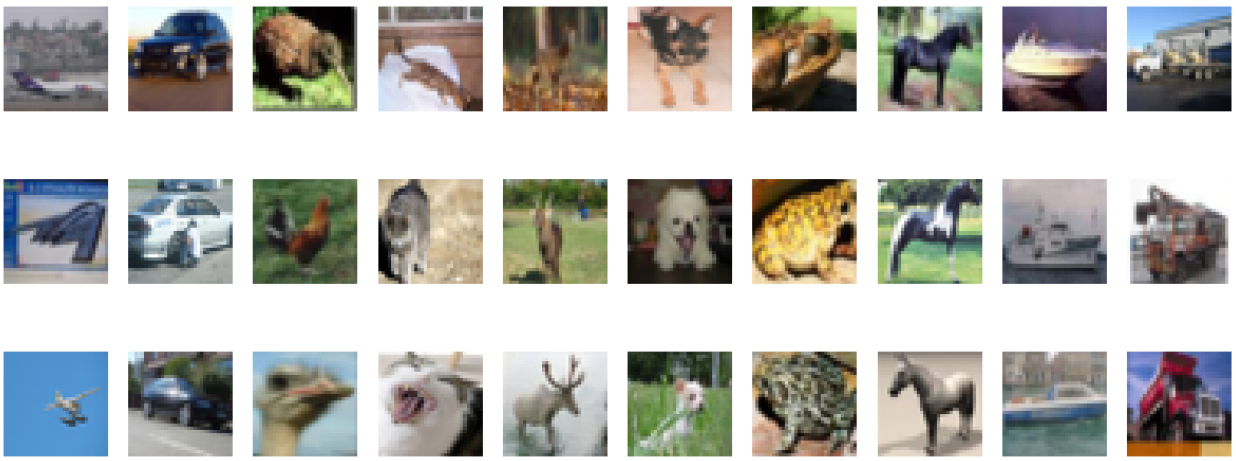


Figure 5: 3x 10 array for cifar10 dataset

5. **Playing with NumPy:**
Write a function called gaussian(n, m) that returns an n × m NumPy array, each entry of which is a random number drawn from the standard normal distribution (Gaussian with mean 0 variance 1). Generate a 10x10 grid of 2-D points using this function and then make a scatter plot of the points using Matplotlib.
Scatter Plot and Code:

Solution:
We can create this gaussian function in a variety of ways with two such appraoches shown below:
Approach 1:

In [6]:
```
import numpy as np
import matplotlib.pyplot as plt

#Creates the array
def gaussian(n: int, m:int):
    """Plot gaussian function

    Args:
    n: initial integer value for the array
    m: initial intger value for the array

    Returns:
    Scatter plot of the n by m by 2 array
    """
    arr=  np.random.randn(n, m, 2)

    #Formats the data
    x, y = arr[:, :, 0].flatten(), arr[:, :, 1].flatten()

    #Plots the points
    plt.scatter(x, y)

    #Adds titles
    plt.xlabel("X")
    plt.ylabel("Y")
    plt.title("100 Gaussian Points")

    #Returns the final result
    return plt.show()


#Calls the data
gaussian(10,10)
```

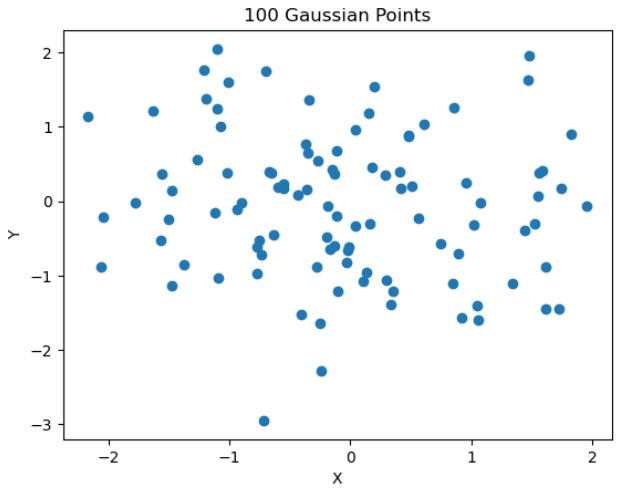


Figure 6: Approach 1

Approach 2:

In [7]:

```python
import numpy as np
import matplotlib.pyplot as plt

#Creates the array
def gaussian(n: int, m:int):
    """Plot gaussian function

    Args:
    n: initial integer value for the array
    m: initial intger value for the array

    Returns:
    Scatter plot of the n by m by 2 array
    """
    #Creates the two arrays
    arrx=  np.random.randn(n,m)
    arry = np.random.randn(n,m)

    #Plots the points
    plt.scatter(arrx, arry)

    #Adds titles
    plt.xlabel("X")
    plt.ylabel("Y")
    plt.title("100 Gaussian Points")

    #Returns the final result
    return plt.show() #plt.show()


#Calls the data
gaussian(10,10)
```

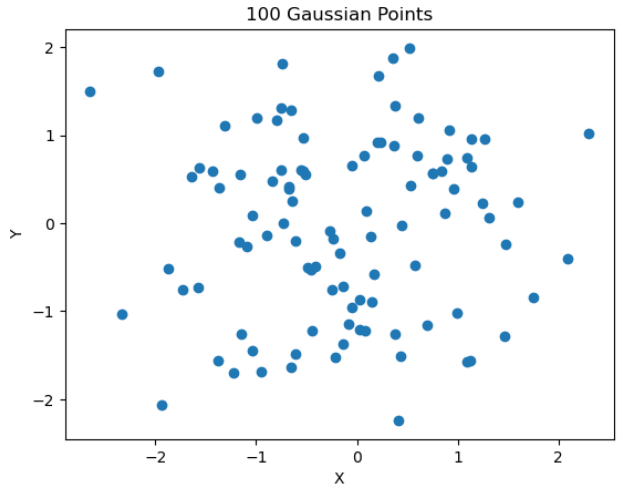


Figure 7: Approach 2

Disclaimer: The base code was written by me and then improved by Chatgpt to be more efficent/look more elegnat. Work was then consulted/compared with Marcelo Fuentes, Michelle Wang, Nick Pacia, Renata Anastasia, Alejandro Castellano and Phil Parvaneh.

Additionally, the latex template and way to display code was carried over from ORIE: 5735 Graph Based Data Science.