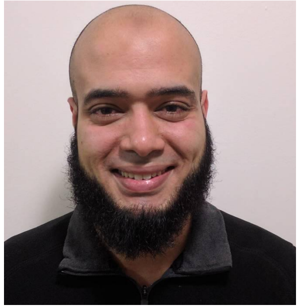
Machine Learning
Logistic Regression

Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / MSc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

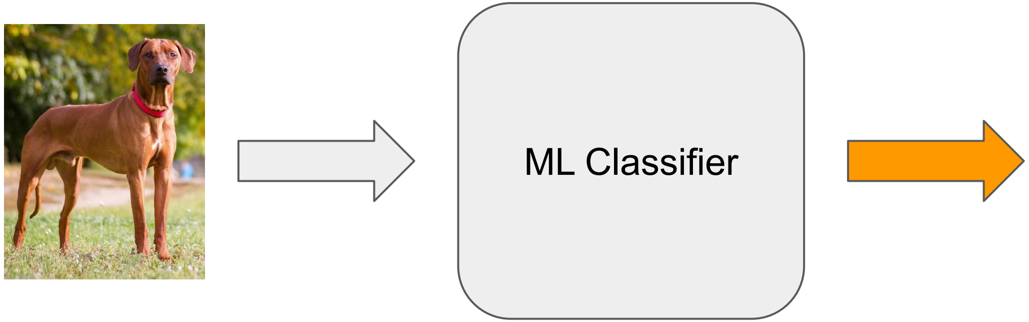


Take Home Messages

- Linear regression shouldn't be used for classification for 2 reasons:
  - Its output range is open [-OO, OO]
  - Linear regression just tries to fit a line, but we need something that separate the classes
  - We need an approach that can handle these limitations
- Sigmoid
  - Differentiable function
  - Its input is called logit in range [-OO, OO]
  - Its output in range [0, 1] ⇒ interpret as probability
  - Logit is proportional to the probability

Logistic Regression

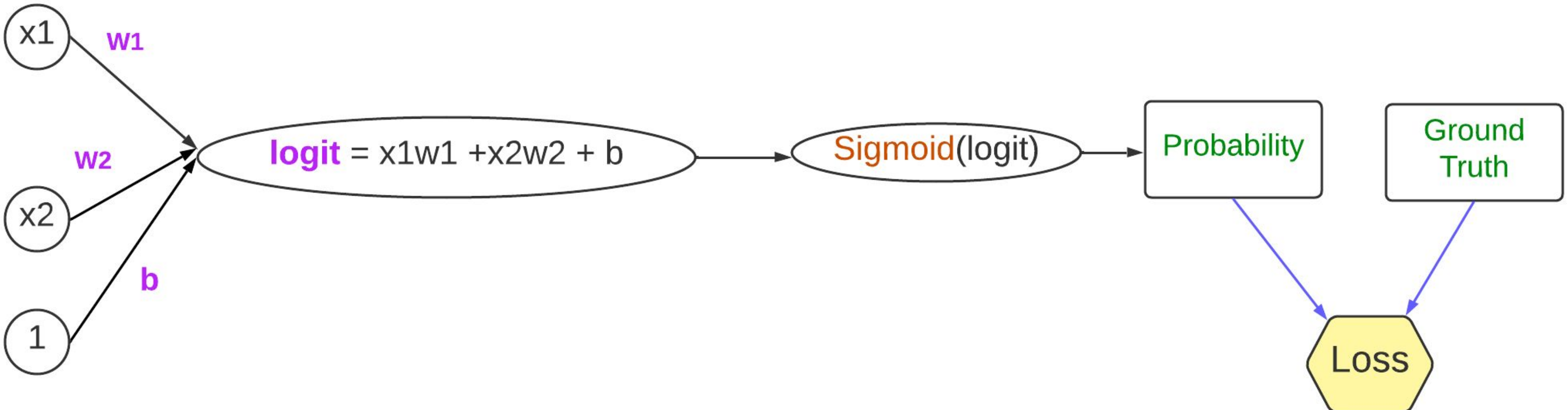
- Logistic Regression is a technique that **predicts the probability** of a binary category P(output | input)
    - For example: Given an image, the probabilities are: dog: 0.8 and cat = 0.2 *(1 - 0.8)*
- With a threshold, we can convert this probability to a discrete category
    - For example, with threshold = 0.6, the image is dog category



- This is a **dog** (0.8 probability)
    - 0.8 > 0.6 (threshold)
- This is a cat (0.2 probability)

From Linear Regression to Logistic Regression

- We extend the linear regression with a non-linear squashing to range [0, 1] using the sigmoid function
- This output probability:
    - With the ground truth + loss function ⇒ we can use to optimize a model
    - With a threshold, it can be converted to a discrete label

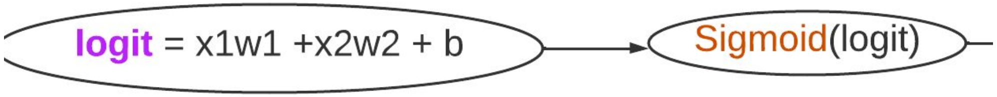


The Decision Boundary

- Now, we solved the output range problem
  - [0, 1] range
- What kind of geometry does our solution represent?
- Assume we applied a threshold of 0.5 to the output probability
  - E.g. dog = 0.85 probability, then as 0.85 >= 0.5, we classify as dog
- What kind of decision boundary we build?

The Decision Boundary

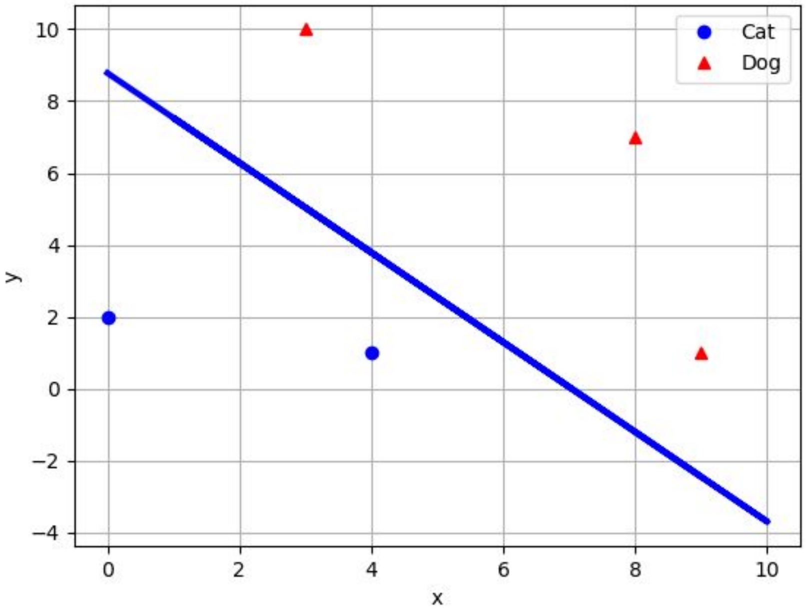
- Given threshold p = 0.5, then we classify as 1 if p >= 0.5
  - p = sigmoid(logit)
- sigmoid(logit) >= 0.5
  - Which logit has such probability? Logit = 0
- Then sigmoid(logit) >= 0.5 is as same as logit >= 0
  - Recall: Both curves are increasing functions from sigmoid lecture
- Hence, if x1w1 + x2w2 + b >= 0, then we classify as 1
- In other words, x1w1 + x2w2 + b acts as **line separator** between 2 classes



Probability	Logit
0.5	0
0.7	0.85
0.9	2.2

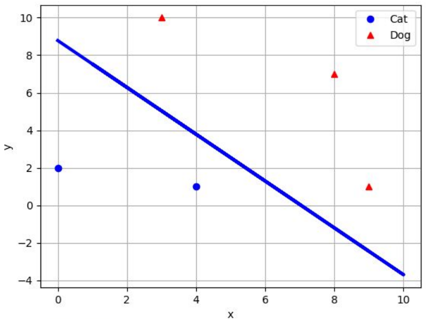
The Decision Boundary

- Now we learn a line that splits the data rather than fitting it
- In theory, we can also use non-linear boundaries
  - E.g. an ellipse, where points inside it are zero and outside are 1



Drawing the Decision Boundary

- Our decision is boundary is the line x1w1 + w2x2 + b
  - For p = 0.5, logit = 0. So we need 2 points with x1w1 + w2x2 + b = 0 (on the line)
- Let's draw it
  - Let's derive its mx + c line (recall: m = (y2-y1) / (x2-x1))
- Let's try x1 {0, and 1} and derive x2
  - x1w1 + w2x2 + b = 0
  - x2 = (-x1w1 - b) / w2
  - At x1 = 0, then x2 = -b / w2
    - Notice, this x2 is the intercept, so it is **c**
  - At x1 = 1, then x2 = (-w1 - b) / w2
  - Then m = dy/dx = -w1 / w2
- So overall line is **[-w1 / w2]**x + **-b / w2**



Why Sigmoid? Any Other Choices?

- Sigmoid solved our 2 problems
- However, one more important thing is being able to use for optimization
    - Sigmoid is differentiable everywhere
    - Sigmoid works fine in shallow network
    - Sigmoid works fine in the output layer
- Can we use something else?
    - In theory yes. But we need to take care of the previous 3 concerns
        - Output range + Meaningful decision boundary + friendly with optimizers
    - Note: some algorithms’ output is still open range, such as SVM

Data

```python
# In practice, we scale input data
if True:    # linearly separable
    X = np.array([[0, 2], [4, 1], [3, 10], [9, 1], [8, 7]])
    y = np.array([0, 0, 1, 1, 1])
else:       # NOT linearly separable
    X = np.array([[0, 2], [3, 10], [9, 1], [8, 7]])
    y = np.array([0, 1, 1, 0])
```

Learn

```
model = linear_model.LogisticRegression().fit(X, y)
# SciKit uses default threshold = 0.5
y_pred = model.predict(X)
y_prob = model.predict_proba(X)

print('ground', y)
print('predct', y_pred)
print(y_prob[:, 1])      # P(c=1|I): probability of being 1
print(f'{np.count_nonzero(y_pred == y)} out of {y.size}')
```

```
ground [0 0 1 1 1]
predct [0 0 1 1 1]
[0.021743   0.1726425  0.94183974 0.87368758 0.99006458]
5 out of 5
```

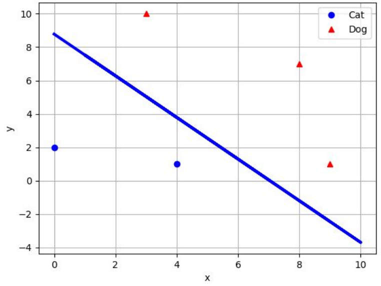
Draw

```
# Get parameters of mx+c equation
b = model.intercept_[0]
w1, w2 = model.coef_.T
m, c = -w1 / w2, -b / w2    # won't work if w2 is zero

plt.plot(X, m * X + c, color="blue", linewidth=3)   # decision bounda

# get cat points and draw them
indices_cat = np.argwhere(y == 0).reshape(-1)
X_cat = np.array([X[idx] for idx in indices_cat])
plt.plot(X_cat[:, 0], X_cat[:, 1], 'o', color='b', label="Cat")

indices_dog = np.argwhere(y == 1).reshape(-1)
X_dog = np.array([X[idx] for idx in indices_dog])
plt.plot(X_dog[:, 0], X_dog[:, 1], '^', color='r', label="Dog")
```



Question!

- Assume we have 2 points p1 and p2
- We trained a linear regression and logistic regression
- Where is the line of each one?

- For linear regression, the line passes with the 2 points
- For logistic regression it splits them

Logistic Regression Notes

- Logistic Regression is used for classification
  - True statement: probability + threshold ⇒ Classification
- Logistic Regression is a classification model
  - It is fine, but better use word “used for”: LR regresses on probabilities
  - Layer before the generated probabilities are called **logits** (log-odds)
  - From this we get the name: logistic, hence it is called Logistic Regression
- Logistic Regression is a predictive model for discrete output values
  - This is correct, however this is a **special case** (although the most common)
  - Logistic Regression is a predictive model where the output values are a probability distribution (sums to 1) - this is the **general** case
- Logistic Regression is a special case of Generalized Linear Models (GLM) with *logit* as the *link function*

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."