CSCI 495 Loan application Project

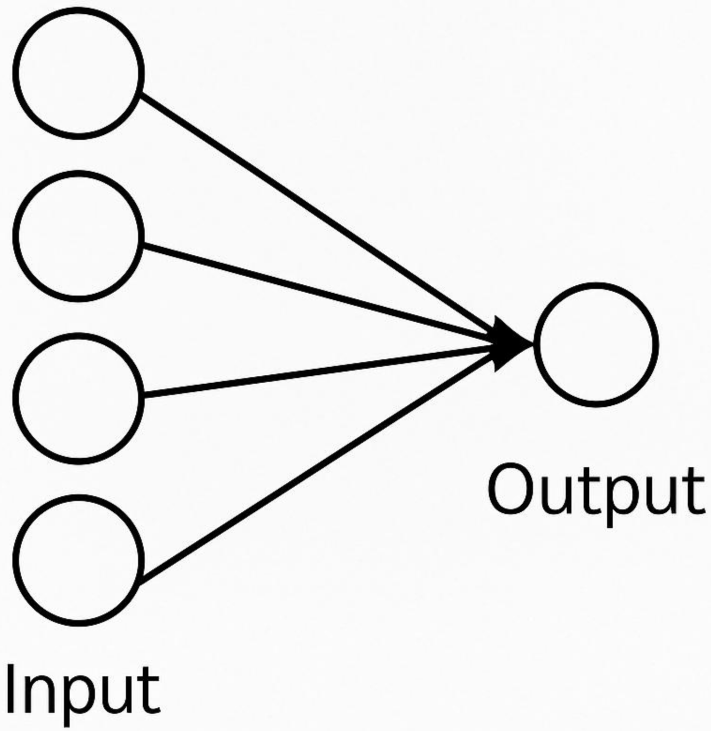
Task assignments among team members

- **Member 1**: Working on importing and exploring the dataset so we can effectively preprocess the data for model training.
- Member 2: Description of the dataset and why we chose it for our project.
- Member 3: Set-up of the project, explaining hardware/software used as well as the evaluation of results through tracked metrics and how they are tracked.
- Member 4: Devleop Models related to research questions and explain how they will be used to get results in the project.

Research Questions

1. How does the performance of a deep neural network compare to traditional machine learning models?
2. How does the performance of a deep neural network compare to shallow neural networks?

LOGISTIC REGRESSION



If your input features are:

- `Income`
- `Credit Score`
- `Loan Amount`
- `Employment Status`
- `Age`

Then logistic regression forms a weighted combination:

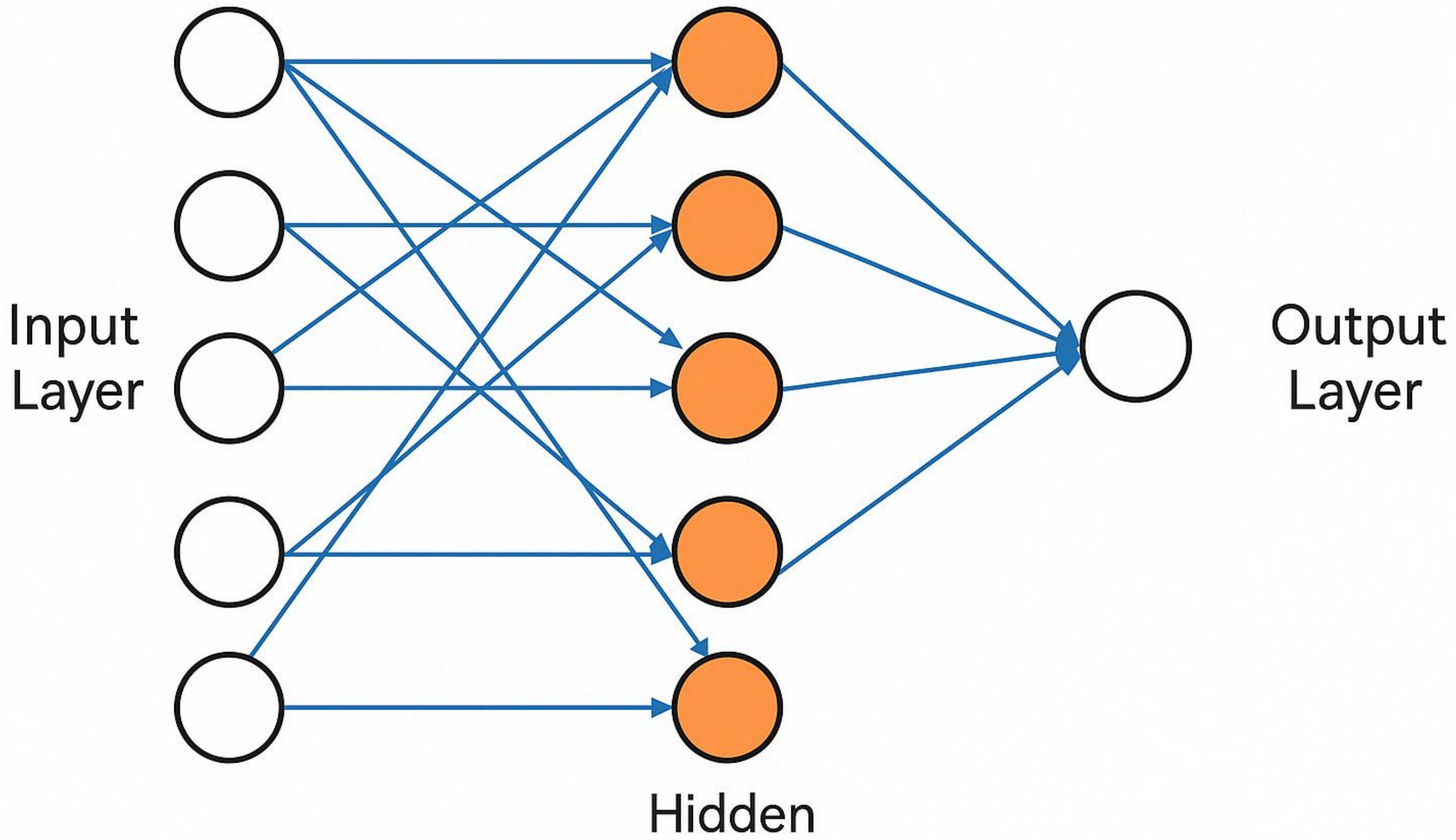
$$z = w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n + b$$

Then passes it through a **sigmoid function**:

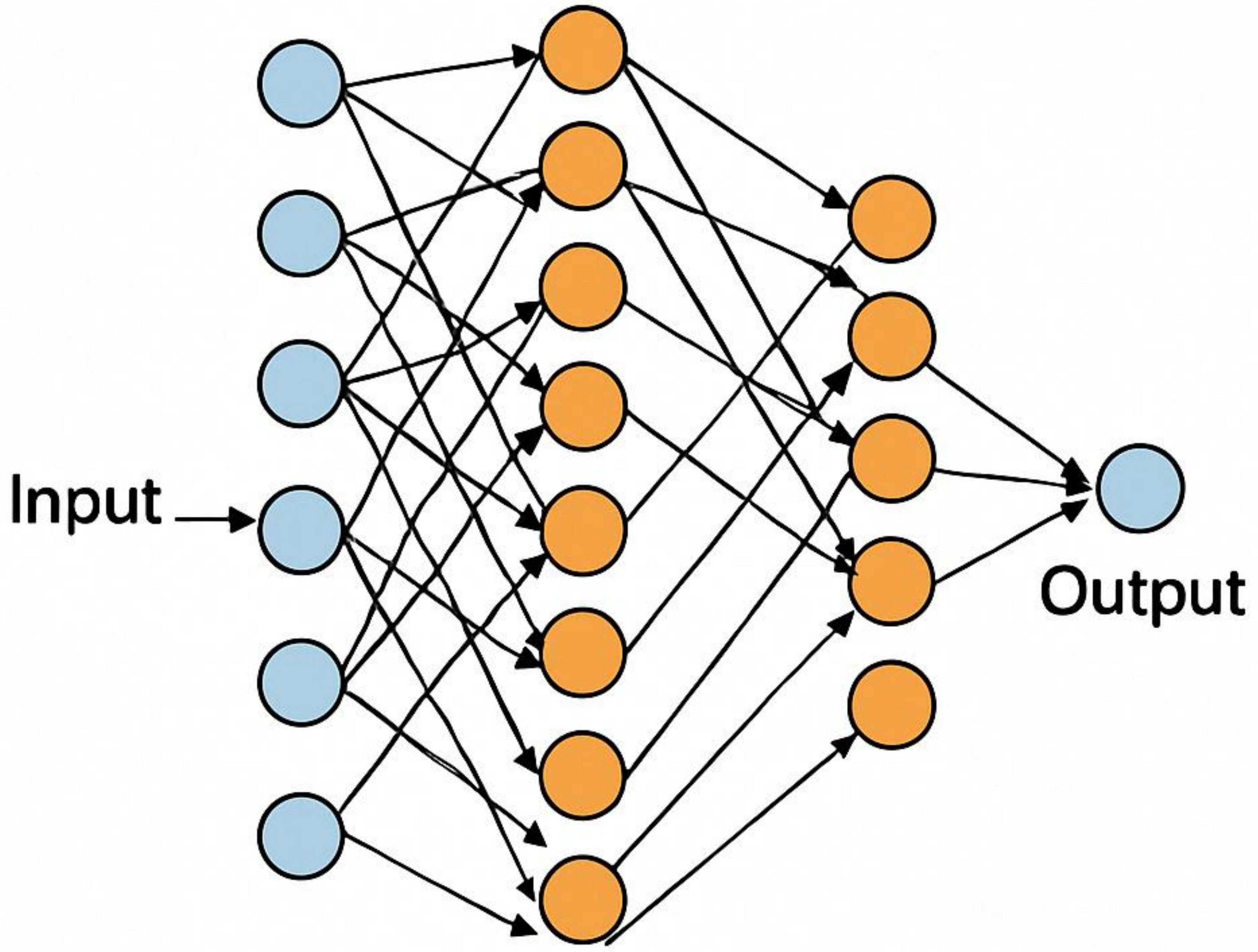
$$\text{Output} = \frac{1}{1 + e^{-z}}$$

This outputs a probability between 0 and 1 — perfect for binary classification (e.g., approve vs reject).

Shallow Neural Network



Deep Neural Network



Dataset Preprocessing

- Improves data quality and compatibility
  - Removes irrelevant data points
  - Reduces noise
  - Convert some features into numerical formats
  - Allows the model to focus on the most important features
- Feature engineering
  - Creates new features for the model to analyze
  - Can uncover hidden relationships between features
  - Can transform existing features into ones that are better suited for the problem
  - Can combine features to reduce model complexity and the potential of overfitting
- Providing the best data possible will decrease the risk of underfitting and give the models a fair chance in testing

Dataset Preprocessing

- The dataset contains a number of categorical columns that will be encoded
  - Ordinal: checking status, savings status, employment, own telephone, and foreign worker
  - One hot: job, housing, other payment plans, property magnitude, other parties, personal status, credit history, purpose
- Ordinal encoding turns categories into numbers
  - This allows the model to interpret the inherit order of the categories
- One hot encoding turns each unique category into its own column
  - The selected category for an entry will receive a 1, and the others will receive a 0
  - Prevents the model from trying to find nonexistent relationships based on the category order
  - Can create higher dimensionality and longer training times since it adds lots of new columns (thus, contributing to model complexity and decreasing performance)

Dataset Preprocessing

- Some columns also have distributions with long tails
  - We can replace them with their square roots or logs to eliminate the tails
- Can shift or constrain the range of data points
  - Helps the model better understand the data
  - Prevents bias toward certain features
  - Reduces the effect of outliers
- Standardization:
  - Changes spread of the data
  - Less sensitive to outliers
  - Good for roughly normal data distributions
  - Centers the data around 0, does not constrain the data to a specific range
- Normalization:
  - Does not change the spread of the data, just rescales it
  - Very sensitive to outliers
  - Good for distance-based algorithms
  - Constrains data between a specific range

Dataset Preprocessing

- Ratio columns
  - Creates new features by computing the ratio of two other features
- Custom columns:
  - Creates new features using an algorithm that combines existing features
- Bucketizing:
  - Sometimes placing data points into buckets and transforming them into categories can help
  - Has the potential to reduce overfitting
- Potential examples:
  - monthly credit burden = credit amount / duration
  - installments per credit = installment commitment / credit amount
  - credit_per_dependent = credit_amount / num_dependents
  - young high credit = age <= 25 and credit amount > 5000
  - bucketized age = bucketize (age, [25, 39, 59, 89, 150])
- Feature selection algorithms will be used to test the features and decide which ones work well without overfitting the tested models
  - Correlation heatmaps can also help identify useful features

Dataset Description – German Credit Data

- Source: UCI Machine Learning Repository
- Author: Dr. Hans Hofmann
- Size: 1,000 Instances, 21 Features
- Target Variable: class – Good vs. Bad Credit Risk

Why This Dataset?

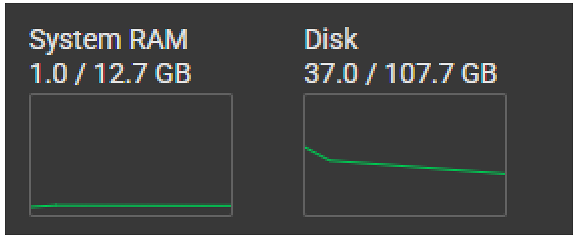
- Real-World Relevance: Simulates real credit approval scenarios.
- Feature Variety: Includes both financial and personal data.
- Binary Classification: Suited for sigmoid-based DNN output.
- Cost-Sensitive Learning: Dataset has a built-in cost matrix for evaluating risk

How the Dataset Supports Our Goal :

- Can a DNN accurately predict loan approvals from applicant data?
- Clean, labeled historical data for model training.
- Binary output matches our model structure.
- Reflects real-world financial decision-making.
- Enables cost-aware evaluation of model predictions

Project Set-Up

- Hardware
  - T4 GPU
- Software
  - Google Colab *(Cloud-based Python notebook)*
  - Python
    - TensorFlow
    - sklearn
    - numpy
    - pandas
    - matplotlib



Project Result Evaluation

- How will we know our goals are met?

1. *How does the performance of a deep neural network compare to traditional machine learning models?*
2. *How does the performance of a deep neural network compare to shallow neural networks?*
3. *Determining if and how well a DNN can accurately predict loan approvals from applicant data.*

- Metrics
  - How easily it processes complex data
  - Time
  - Accuracy

END