**Week-1 Class Summary**

**1. Course Introduction**

* Reviewed the **syllabus** to get things started.
* Mentioned that lecture notes and slides will be posted as we progress.

**2. Neural Network Motivation (TensorFlow Playground)**

* Demonstrated the [TensorFlow Playground](https://playground.tensorflow.org/) to show how neural networks perform **binary classification** (blue vs. orange points) across four example datasets.
* Observed how **hidden layers** make the network more powerful by enabling it to learn **nonlinear decision boundaries** (e.g., inner vs. outer circles).
* Key takeaway:
  + **Linear decisions** = weighted sum of inputs.
  + **Nonlinear decisions** = require hidden layers.

**3. Weka (Using the Iris Dataset)**

* **Purpose in Class**:  
  We used Weka as a teaching tool to **motivate concepts** such as train–test splits, histograms, pairwise plots, and evaluation. Students don’t need to install Weka—later we’ll reimplement these ideas directly in **Python code** rather than with the Weka GUI.
* **Dataset: Iris (UCI Machine Learning Archive)**
  + **Features**: 4 dimensions — sepal length, sepal width, petal length, petal width.
    - Thought experiment: instead of just 4 features, imagine using **flower photos** of size 100×100 pixels in color → **100 × 100 × 3 = 30,000 input features**.
    - This highlights the importance of **feature engineering**:
      * For the **XOR dataset**, adding *x1 × x2* helps.
      * For the **two-circles dataset**, adding *x1²* and *x2²* helps.
      * For **images** (like MNIST digits), engineered features (e.g., stroke counts – as simple as how many vertical and horizontal lines exist in the image) can help, though hidden layers in neural networks often learn better transformations automatically.
  + **Class labels**: 3 categories — setosa, versicolor, virginica.
  + **Examples**: 150 total data points.
* **Exploratory Analysis in Weka**
  + Plotted **histograms** for each feature.
  + Created **pairwise scatter plots**, adding jitter and increasing point size for clarity.
* **Running Classifiers in Weka**
  + **J48** (decision tree).
  + **IBk** (k-nearest neighbors under “lazy”).
* **Evaluation Methods**
  + **Train/test split** (e.g., 90% train, 10% test).
  + More examples for training is better for accuracy in general, but what if the test set is not comprehensive (because it is small), so it is good to run multiple times. That leads us to **10-fold cross validation** as a better compromise.

**4. Precision & Recall**

* Introduced **precision** and **recall**, computed per class using a **3×3 confusion matrix**.
* Learned to calculate:
  + Precision = TP / (TP + FP).
  + Recall = TP / (TP + FN).
* Discussed averaging strategies:
  + **Macro-average**: unweighted mean across classes.
  + **Weighted average**: accounts for class sizes (preferred in many cases).

**Practice Question (Imbalanced Classes)**

Rows = **true labels**, columns = **predicted**.

|  | **Pred A** | **Pred B** | **Pred C** |
| --- | --- | --- | --- |
| **True A (100)** | 80 | 15 | 5 |
| **True B (30)** | 10 | 15 | 5 |
| **True C (20)** | 2 | 6 | 12 |

1. Compute accuracy.
2. Compute **Precision** and **Recall** for each class (A, B, C).
3. Compute the **Macro average** (unweighted mean) and the **Weighted average** (weighted by support = number of true examples per class) for Precision and Recall.

**Solution**

**Class totals**

* Supports (true counts): A=100, B=30, C=20 (total **150**)
* Column totals (predicted):
  + Pred A = 80+10+2 = **92**
  + Pred B = 15+15+6 = **36**
  + Pred C = 5+5+12 = **22**

**Accuracy**

**Accuracy** = (sum of diagonal) / (total)  
= (80 + 15 + 12) / 150  
= **107 / 150 ≈ 0.713** → **71.3%**

**Weighted recall (by support)**equals**accuracy** (see below).

**Per-class metrics**

(Precision = TP/(TP+FP); Recall = TP/(TP+FN))

* **Class A**
  + TP = 80, FP = 92−80 = 12, FN = 100−80 = 20
  + Precision = 80/92 ≈ **0.870**
  + Recall = 80/100 = **0.800**
* **Class B**
  + TP = 15, FP = 36−15 = 21, FN = 30−15 = 15
  + Precision = 15/36 ≈ **0.417**
  + Recall = 15/30 = **0.500**
* **Class C**
  + TP = 12, FP = 22−12 = 10, FN = 20−12 = 8
  + Precision = 12/22 ≈ **0.545**
  + Recall = 12/20 = **0.600**

**Macro averages (simple mean across classes)**

* **Macro Precision** = (0.870 + 0.417 + 0.545) / 3 ≈ **0.611**
* **Macro Recall** = (0.800 + 0.500 + 0.600) / 3 ≈ **0.633**

**Weighted averages (by support: 100, 30, 20; total 150)**

* **Weighted Precision** = (100·0.870 + 30·0.417 + 20·0.545) / 150 ≈ **0.736**
* **Weighted Recall** = (100·0.800 + 30·0.500 + 20·0.600) / 150 ≈ **0.713 (equals accuracy)**