AI Assignment No. 2

Supervised Learning

Group_A1_17
Tomás Oliveira - up202208415
Diogo Ferreira - up202205295
Álvaro Torres - up202208954

Dataset: Steel Plate Defect Prediction

- Dataset used was generated from a deep learning model trained on the Steel Plates Faults dataset from UCI.
  - Files:
    - train.csv - the training dataset; there are 7 binary targets: Pastry, Z_Scratch, K_Scatch, Stains, Dirtiness, Bumps, Other_Faults
    - test.csv - the test dataset;
    - sample_submission.csv - a sample submission file in the correct format
- The original dataset of steel plate faults is classified into 7 different types. The goal was to train machine learning for automatic pattern recognition.
- Dataset has no missing values.
- **Target variables are 7 types of defects: Pastry, Z_Scratch, K_Scatch, Stains, Dirtiness, Bumps and Other_Faults.**
- **Supervised learning problem: our objective is to predict the probability of each of the 7 binary targets on a given random steel plate that has 27 features.**

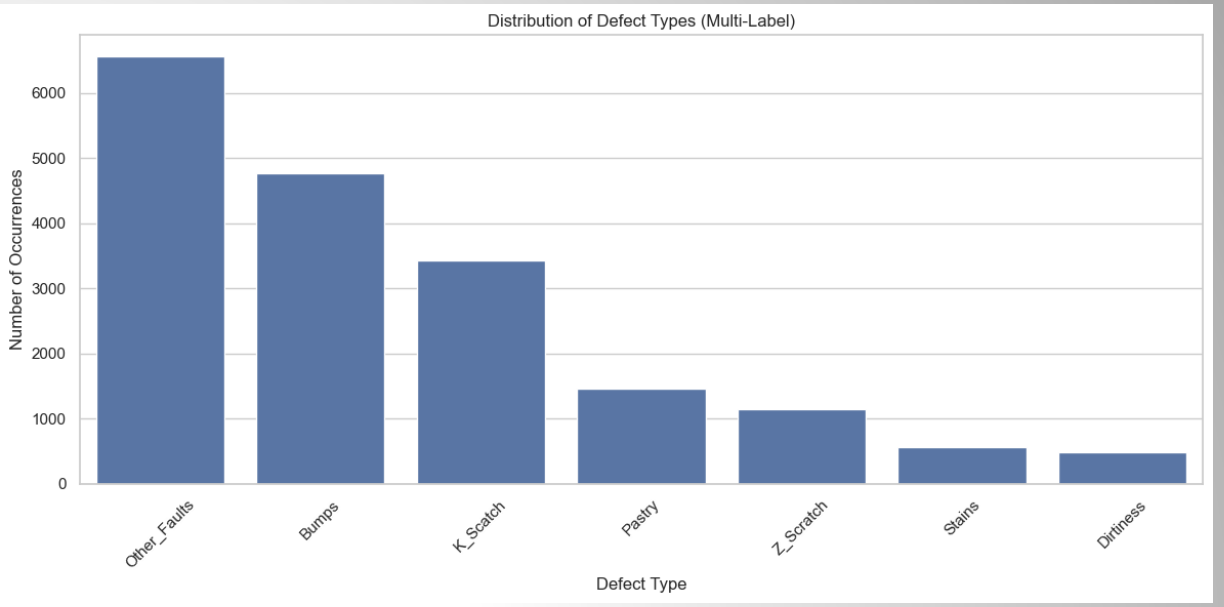
Related Work

- Dataset used: Steel Plate Defect Prediction | Kaggle
- Original dataset: Steel Plates Faults - UCI Machine Learning Repository
- Recent codebases:
  - ravenous_steel
  - Steel Plate Defect Prediction 0.88838 AUC Score

Methodology

- **Programming language:** Python;
- **Libraries and tools:**
  - Pandas to load training dataset into memory and for data manipulation;
  - Numpy for numerical operations;
  - Matplotlib and Seaborn for visualizations (plots, histograms and heatmaps);
  - scikit-learn for data splitting and data preprocessing standardization;
  - joblib to store scaler (data preprocessing standardization) and models;
  - GridSearchCV for hyperparameter tuning (finding optimal hyperparameter configuration with the best performance);
- **Models/Algorithms:**
  - Decision Tree: DecisionTreeClassifier (One-Vs-Rest) (scikit-learn);
  - k-Nearest Neighbors (k-NN): KNeighborsClassifier (scikit-learn);
  - Multi-Layer Perceptron (MLP (Neural Network): MLPClassifier (scikit-learn);
  - Linear Support Vector Classifier (SVM) with One-vs-Rest: LinearSVC (scikit-learn);

Data Preprocessing



- **Exploratory Data Analysis:**
  - Read data;
  - Explored variables:
    - Data types and non-null counts analyzed;
    - Missing values per column counted;
    - Statistical Summary for Numerical Features;
    - Target Variable Distribution (distribution of defect types);
      - Number of samples with no defects;
    - Visualizations for Features;
      - Histograms for numerical features;
      - Correlation Heatmaps for feature-feature and feature-target;
    - Initial Preprocessing Considerations (Based on EDA) → Conclusions for data preprocessing step;
- **Train dataset splitting** into features (X) and targets (y) with respective training (80%) and validation (20%) subsets;
- **Data preprocessing (based on results from EDA):**
  - Performed feature scaling (standardization) on features subsets using StandardScaler from scikit-learn:
    - First fitted only on X_train (to avoid data leakage), then transformed both training and validation data (X_train and X_val);
      - fit(X) → calculates statistics (mean, std) and learns the parameters; transform(X) → applies:$(X-mean)/std$(X−mean)/std for each feature to scale the data;
    - y_train, y_val stay unchanged;
    - Scaler saved for later use on test data;
    - After standardization, all features have mean ~0 and standard deviation ~1, confirming successful scaling;

Dataset	Features Shape (X)	Target Shape (y)
Training (_train)	(15375, 27)	(15375, 7)
Validation (_val)	(3844, 27)	(3844, 7)

Data split into Training and Validation sets

Machine Learning (SL) Models

- **Decision Tree Classifier (with OneVsRestClassifier)**
  - Algorithm: Non-parametric tree-based classifier for multi-label classification
  - Key Parameters: max_depth=15, max_leaf_nodes=100, class_weight='balanced', criterion='gini/entropy'
  - Justification: Easy to interpret, handles multi-label directly, balanced weights to address class imbalance in steel defect data
- **K-Nearest Neighbors (k-NN)**
  - Algorithm: Instance-based learning for multi-label classification
  - Key Parameters: n_neighbors=3-15, weights='uniform/distance', metric='euclidean/manhattan/minkowski'
  - Justification: Non-parametric, naturally handles multi-label outputs, effective with scaled steel plate features
- **Multi-Layer Perceptron (MLP) Neural Network**
  - Algorithm: Feedforward neural network with backpropagation
  - Key Parameters: hidden_layers=(100,50), activation='relu/tanh', solver='adam', early_stopping=True
  - Justification: Captures non-linear relationships in steel defect features, handles multi-label natively, robust with 27 input features
- **Linear Support Vector Machine (LinearSVC with OneVsRestClassifier)**
  - Algorithm: Linear SVM optimized for large datasets
  - Key Parameters: C=0.1-10.0, class_weight='balanced', dual=False, max_iter=5000
  - Justification: Fast training/prediction for 19,219 samples, linear decision boundaries suitable for scaled features, OneVsRest handles 7 defect types
- **Model Selection Strategy**
  - GridSearchCV with 3-fold cross-validation for hyperparameter tuning
  - Scoring metric: F1-samples (appropriate for multi-label imbalanced data)

Evaluation and comparison of models

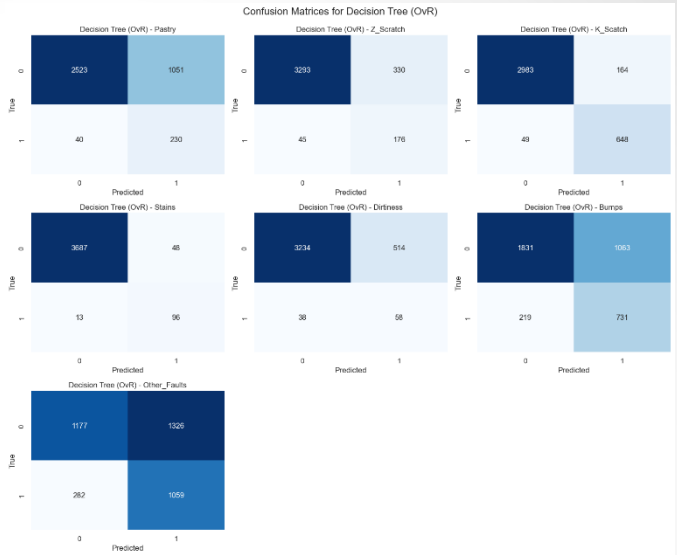
- **MLP Classifier: Top Overall Performer**
  - Highest Subset Accuracy: **0.4451** (all 7 labels correct for ~44.5% of samples).
  - Lowest Hamming Loss: **0.1109** (smallest fraction of wrong labels).
  - Competitive F1-Score (Samples): **0.4341**.
  - Trade-off: Second longest effective Train Time (880.7s for GridSearchCV).
- **k-NN: Close Competitor, Time Intensive**
  - Subset Accuracy: **0.4373** (very close to MLP).
  - F1-Score (Samples): **0.4272**.
  - Good Hamming Loss: **0.1284** (2nd best).
  - Longest Train Time (3589s) and slowest Test Time (5.45s).
- **Decision Tree (OvR) with class_weight='balanced':**
  - Highest F1-Score (Samples): **0.5628** and Recall (Samples): **0.7794**.
  - Lower Subset Accuracy: **0.2500**.
  - Higher Hamming Loss: **0.1926**.
  - Moderate Train Time (366s). Best Params: max_depth: 15, max_leaf_nodes: 100.
- **Linear SVC (OvR) with class_weight='balanced':**
  - Similar to Decision Tree: High F1 (Samples) **0.5365** and Recall **0.769**.
  - Lowest Subset Accuracy: **0.2190**.
  - Higher Hamming Loss: **0.2140**.
  - Strength: Fast Test Time (**0.005s**).
  - Moderate Train Time (790s).

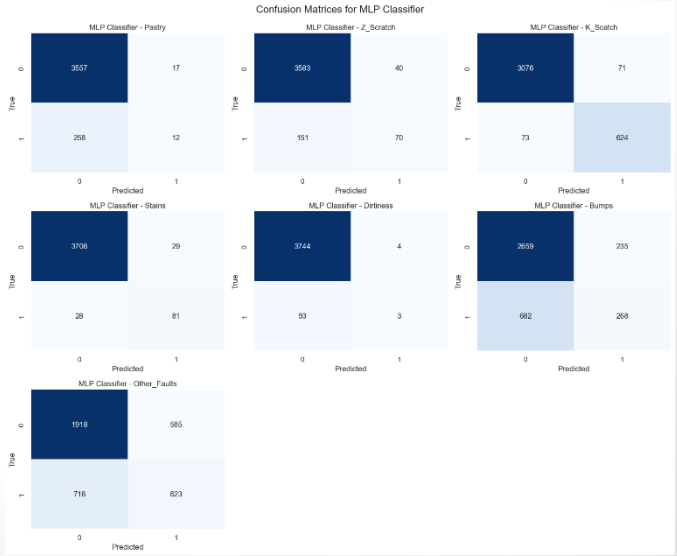
- **Key Takeaways:**
  - MLP excels in overall correctness (Subset Accuracy and Hamming Loss).
  - Decision Tree and Linear SVC (balanced) show inflated sample F1/Recall.
  - k-NN is good but very slow in the testing phase phase for this dataset.
  - Clear trade-off: Performance vs. Computational Cost.

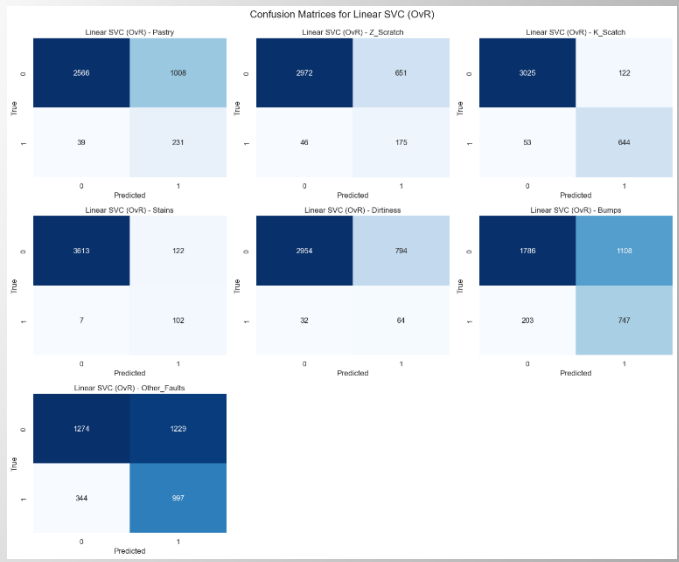
	Subset Accuracy	Precision (Samples)	Recall (Samples)	F1-Score (Samples)	Hamming Loss	Train Time (s)	Test Time (s)	Best Params
Decision Tree (OvR)	0.250000	0.475217	0.779396	0.562764	0.192582	366.188659	0.013024	{'estimator__class_weight': 'balanced', 'estimator__criterion': 'gini', 'estimator__max_depth': 15, 'estimator__max_features': None, 'estimator__max_leaf_nodes': 100, 'estimator__min_samples_leaf': 1, 'estimator__min_samples_split': 5, 'estimator__splitter': 'best'}
k-NN	0.437305	0.427419	0.427029	0.427159	0.128438	3588.996321	5.454489	{'algorithm': 'auto', 'leaf_size': 10, 'metric': 'euclidean', 'n_neighbors': 3, 'p': 1, 'weights': 'uniform'}
MLP Classifier	0.445109	0.433012	0.436785	0.434096	0.110896	880.739357	0.051109	{'activation': 'tanh', 'alpha': 0.001, 'batch_size': 64, 'early_stopping': True, 'hidden_layer_sizes': (100, 50), 'learning_rate_init': 0.01, 'max_iter': 500, 'n_iter_no_change': 10, 'solver': 'adam'}
Linear SVC (OvR)	0.219043	0.445543	0.769511	0.536516	0.213988	789.829520	0.005462	{'estimator__C': 10.0, 'estimator__class_weight': 'balanced', 'estimator__dual': True, 'estimator__loss': 'squared_hinge', 'estimator__max_iter': 5000, 'estimator__tol': 1e-05}

Confusion Matrices

- **General Pattern:** High True Negatives (TNs) for most defects (expected, as defects are often absent). Key differences are in TP, FP, FN.
- **Decision Tree (OvR) (class_weight='balanced'):**
  - Higher True Positives (TPs) due to balancing (e.g., Pastry: 230 TPs).
  - Increased False Positives (FPs) as a trade-off (e.g., Bumps: 1063 FPs).
  - Over-predicts rare defects (Stains, Dirtiness) leading to substantial FPs.
  - Other_Faults (most common): Matrix appears "balanced" among TN, FP, TP.
- **k-NN (More Conservative):**
  - Lower FPs generally vs. balanced DT.
  - Decent TPs for common defects (Other_Faults: 489, Bumps: 353).
  - High False Negatives (FNs) for common defects (Other_Faults: 852, Bumps: 597) - misses them often.
  - Other_Faults matrix: TN is highest.
- **MLP Classifier (Best Overall Balance):**
  - Good TP/FP balance for common/moderate defects (e.g., K_Scatch: 624 TPs vs. 71 FPs).
  - Still shows higher FNs for most frequent defects (Other_Faults, Bumps), similar to k-NN.
  - Interesting: K_Scratch (3rd most common) predicted best (high TPs).
- **Linear SVC (OvR) (class_weight='balanced'):**
  - High TPs (e.g., Pastry: 231 TPs).
  - Significant FPs across types (e.g., Dirtiness: 794 FPs).
- **Common Themes:**
  - Common defects (Other_Faults, Bumps, K_Scratch) generally have more TPs.
  - Predicting rare defects accurately is hard.
  - class_weight='balanced' (DT, Linear SVC) boosts recall/TPs per label but increases FPs and hurts overall (multi-label) precision.

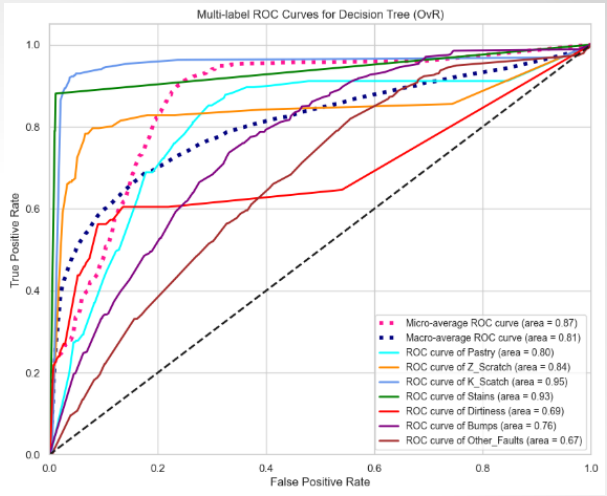


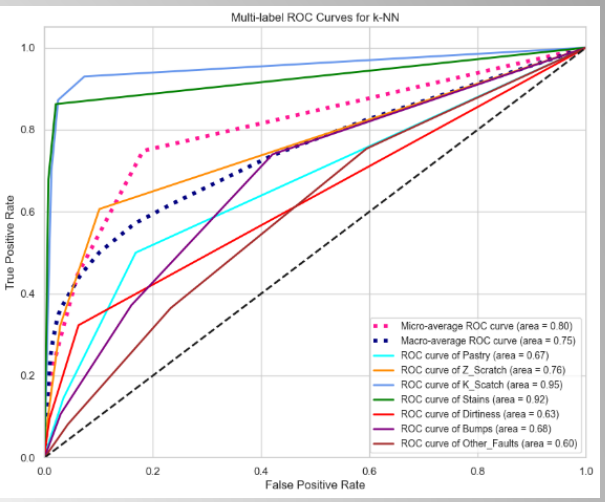


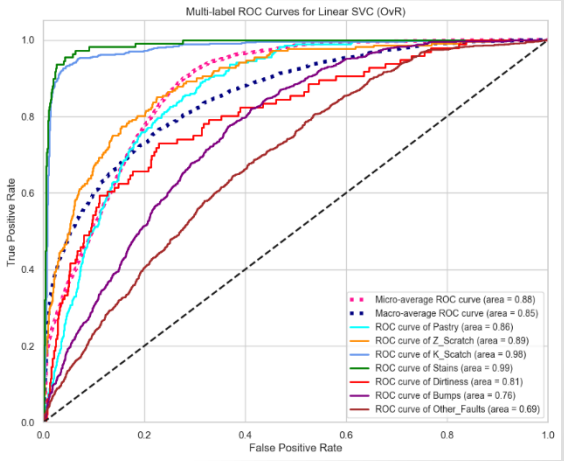


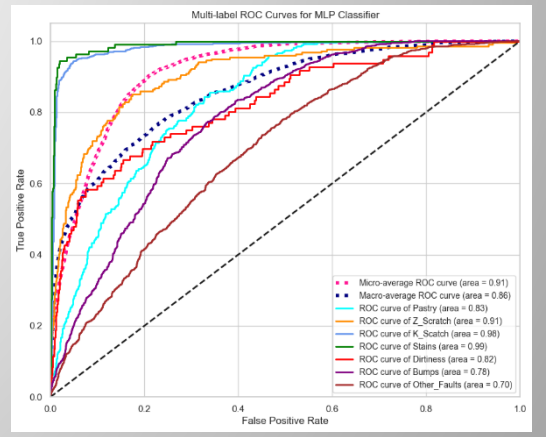
ROC Curves and AUC

- **MLP Classifier: Best Discriminatory Power**
  - Highest Averages: **Macro-AUC: 0.8590, Micro-AUC: 0.9143**.
  - Strong individual AUCs, especially for less frequent defects:
    - K_Scratch: 0.98
    - Z_Scratch: 0.91
    - Stains: 0.99
  - Lowest (but still best) AUCs for most frequent defects: Other_Faults: 0.70, Bumps: 0.78.
- **Linear SVC (OvR): Strong Averages, Balanced Effect**
  - High Averages: **Macro-AUC: 0.8536, Micro-AUC: 0.8756**.
  - Individual AUCs very close to MLP/DT, likely due to class_weight='balanced' affecting probability scores.
- **Decision Tree (OvR): Good Averages, Mixed Individual**
  - Averages: **Macro-AUC: 0.8069, Micro-AUC: 0.8660**.
  - Good AUC for K_Scratch: 0.95.
  - Excellent AUC for Stains (rare): 0.92.
  - Lower AUCs for Dirtiness (rare): 0.69.
  - Surprisingly low AUCs for most frequent defects: Other_Faults: 0.67, Bumps: 0.76.
- **k-NN: Lowest Average AUCs**
  - Averages: **Macro-AUC: 0.7459, Micro-AUC: 0.8045**.
  - Poor AUCs for frequent defects: Other_Faults: 0.60, Bumps: 0.68.
  - Better for some less frequent: Stains: 0.92, K_Scratch: 0.95, Z_Scratch: 0.76.
- **General ROC/AUC Patterns:**
  - MLP consistently best at distinguishing positive/negative instances.
  - Micro-AUC > Macro-AUC generally, suggesting better performance on more common labels (which have more weight in micro-averaging).
  - An interesting pattern:
    - Models with class_weight='balanced' (DT, Linear SVC) and even MLP/k-NN sometimes show higher AUCs for certain rare defects (Stains) than for very frequent ones (Other_Faults, Bumps).
    - This might indicate that while overall prediction for frequent defects is hard (many borderline cases or complex patterns), the distinctness of some rare defect signatures (when found) is picked up well by the probability scores.









Conclusion

- **Goal:** Predict multiple steel plate defects using supervised learning.
- **Top Model: MLP Classifier**
  - Best Subset Accuracy (0.4451), lowest Hamming Loss (0.1109).
  - Best average ROC AUCs (Macro: 0.8590, Micro: 0.9143).
  - Trade-off: Significant training time (881s).
- **Other Model Insights:**
  - **k-NN:** Second on Subset Accuracy (0.4373), good Hamming Loss; very slow train and prediction.
  - **Decision Tree (OvR) & Linear SVC (OvR):**
    - class_weight='balanced' led to high per-sample Recall/F1.
    - Poor Subset Accuracy and high Hamming Loss.
- **Core Challenge: Class Imbalance**
  - All models struggled with infrequent defects (e.g., Stains, Dirtiness).
  - **Confusion Matrices showed:**
    - High FPs for many classes when using class_weight='balanced' (DT, Linear SVC) as models over-predicted positives.
  - **ROC AUCs reflected this:**
    - Models (especially DT, Linear SVC, MLP) surprisingly showed higher AUCs for some rare defects like Stains than for the most frequent ones (Other_Faults, Bumps).
    - This suggests that while rare, these defects might have distinct enough signatures for good probability separation when the model does focus on them (due to balancing or learning capacity).
    - The most frequent defects (Other_Faults, Bumps) had lower AUCs across multiple models, indicating they are harder to cleanly separate based on probability scores, possibly due to more overlapping feature distributions with non-defect instances or other defect types.
- **Multi-Label Complexity:**
  - Predicting all 7 labels perfectly (Subset Accuracy) is tough; best was ~44.5% (MLP).

AI Assignment No. 2

Supervised Learning

Group_A1_17
Tomás Oliveira - up202208415
Diogo Ferreira - up202205295
Álvaro Torres - up202208954