**Final Project Report**

**Interactive RNA-Seq Classifier and DEG Explorer for Heart Disease Using the Magnetique Dataset**

**Abstract**

This project developed an interactive web application for classifying heart disease using RNA-Seq gene expression data from the Magnetique dataset. The pipeline involved data preprocessing, normalization, feature selection using the top 1000 most variable genes, and training various machine learning models (Logistic Regression, Random Forest, Decision Tree, SVM, MLPClassifier) for both binary (Control vs. Dilated Cardiomyopathy) and multi-class (Control vs. DCM vs. Hypertrophic Cardiomyopathy) classification. Differential Expression Analysis (DEG) was performed, and model interpretability was enhanced using SHAP values. The web application, built with Streamlit, provides an accessible interface for data exploration, model evaluation, and interpretation.

**Background and Motivation**

Heart failure represents a significant global health burden. Understanding the molecular signatures distinguishing diseased from healthy tissue can reveal candidate genes and pathways for therapeutic intervention. RNA-Seq technology provides a powerful means to profile gene expression, but extracting meaningful insights requires robust computational analysis. The motivation for this project was to develop a lightweight, interpretable RNA-Seq classifier that allows for fast hypothesis testing and educational use, without requiring access to raw sequencing reads. The Magnetique dataset, focusing on left-ventricular tissue, offered a suitable public resource for this task, aligning with the course's emphasis on interpretable AI in biomedical research.

**Dataset Summary**

**Data Source:** The project utilized the Magnetique dataset (Zenodo:<https://zenodo.org/record/7148045>), described in Sci Data 2023. This dataset comprises RNA-Seq data from left-ventricular tissue of patients with dilated cardiomyopathy (DCM), hypertrophic cardiomyopathy (HCM), and non-failing donor (NFD) hearts.

**Preprocessing:**

1. **Raw Counts:** The initial input was a gene count matrix (gene\_count\_matrix.csv).
2. **Metadata Integration:** Sample metadata (MAGE\_metadata.txt) was integrated to link samples to their respective disease etiologies.
3. **Normalization:** Raw counts were normalized using the Variance Stabilizing Transformation (VST) from the DESeq2 R package (01\_normalize.R). This step accounts for library size differences and stabilizes variance across expression levels, making counts comparable across samples.
4. **Gene Annotation:** Ensembl gene IDs were converted to human-readable gene symbols using the mygene Python library for improved interpretability in plots and reports.

**Splits:** The dataset was consistently split into 70% for training and 30% for testing, ensuring stratified sampling to maintain class proportions in both binary and multi-class scenarios.

**Distributions:** Initial exploratory data analysis (EDA) included checking for missing values and basic descriptive statistics. Principal Component Analysis (PCA) was extensively used to visualize data distributions and assess sample quality.

**PCA Findings and Batch Effect Investigation**

Initial PCA plots (generated in 01a\_Advanced\_PCA\_Analysis.ipynb and run\_final\_analysis.py) consistently showed clear separation of samples based on disease condition (DCM, HCM, NFD). To ensure that this separation was due to biological signal rather than technical artifacts or confounding factors, further PCA plots were generated, colored by metadata attributes such as sex and race.

These investigations confirmed that samples did **not** cluster significantly by sex or race, indicating that these factors were not major drivers of variance in the dataset and suggesting the absence of strong batch effects related to these demographic variables. This validated that the observed separation in PCA was primarily driven by the biological differences between the disease states. A graph with many colored dots

AI-generated content may be incorrect.

**Method Description**

**Workflow:** The project followed a structured workflow:

1. **Data Exploration & Preprocessing:** Initial data loading, normalization, gene annotation, and feature selection.
2. **Feature Selection:** For model training, the top 1000 most variable genes across all samples were selected. This approach reduces dimensionality while retaining genes with significant biological activity, balancing computational efficiency with biological relevance.
3. **Model Training & Evaluation:** Implementation of various ML models for classification, evaluated using standard metrics.
4. **Model Interpretation:** Application of SHAP values to explain model predictions.
5. **Accessibility Wrapping:** Development of an interactive Streamlit web application.

**Models:** The following machine learning models were implemented and evaluated:

* **Logistic Regression:** A linear model serving as a baseline.
* **Random Forest:** An ensemble tree-based model.
* **Decision Tree:** A single tree-based model.
* **Support Vector Machine (SVM):** With a linear kernel.
* **Neural Network (MLPClassifier):** A Multi-Layer Perceptron from scikit-learn.
* **Deep Neural Network (TensorFlow/Keras):** (Implemented and tested separately on Google Colab).

**Evaluation Metrics:** Model performance was assessed using:

* **Classification Report:** Providing precision, recall, F1-score, and support for each class.
* **ROC AUC Score:** Area Under the Receiver Operating Characteristic curve, calculated as macro-average for multi-class problems.
* **Confusion Matrix:** Visualizing true vs. predicted labels.
* **SHAP Values:** Explaining individual feature contributions to predictions.

**Results**

**Differential Expression Analysis (Binary: Control vs. DCM)**

Differential Expression Analysis (DEG) was performed using DESeq2 (R package) comparing Non-Failing Donors (Control) against Dilated Cardiomyopathy (DCM). The analysis identified genes significantly up- or down-regulated between these two groups. The results were saved in DEG\_results.csv, and a volcano plot (volcano\_plot.png) was generated, with the top 10 most significant genes labeled with their gene symbols.

**Summary of Most Significant DEGs:** (Based on DEG\_results.csv and volcano\_plot.png)

**A graph of red dots

AI-generated content may be incorrect.**

**Machine Learning Model Performance**

**Binary Classification (Control vs. DCM)**

Initially, models were trained on a binary classification task (Non-Failing Donor vs. Dilated Cardiomyopathy). Several models, particularly Logistic Regression and SVM, achieved extremely high accuracy (ROC AUC scores near 1.00). This suggested that the two groups are highly separable based on gene expression, potentially indicating a relatively simple classification problem or a strong signal in the data.

**Multi-Class Classification (Control vs. DCM vs. HCM)**

To introduce a more challenging and realistic scenario, the analysis was extended to a 3-class problem, including Hypertrophic Cardiomyopathy (HCM). This significantly increased the complexity, especially given the smaller sample size for the HCM group. The models were trained using the top 1000 most variable genes.

**ROC Curve Comparison:** The multiclass\_roc\_curves\_final.png plot visually compares the macro-averaged ROC curves for all five models. While Logistic Regression and SVM still performed very well (macro AUC ~0.98), the performance for the HCM class was notably lower across all models, highlighting the challenge posed by its limited sample representation.A graph of a multi-class model

AI-generated content may be incorrect.

**Deep Neural Network (TensorFlow on Google Colab)**

Due to local environment compatibility issues, a Deep Neural Network (DNN) model using TensorFlow/Keras was developed and executed on Google Colab, leveraging GPU acceleration. The DNN was configured for binary classification (Control vs. DCM) with the following architecture:

* **Input Layer:** Dense(512, activation='relu', input\_shape=(num\_features,))
* **Hidden Layer 1:** Dropout(0.3)
* **Hidden Layer 2:** Dense(128, activation='relu')
* **Hidden Layer 3:** Dropout(0.2)
* **Output Layer:** Dense(1, activation='sigmoid')
* **Compilation:** optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy', tf.keras.metrics.AUC(name='auc')]
* **Training:** epochs=100, batch\_size=32, validation\_split=0.2, EarlyStopping callback.A screenshot of a computer

  AI-generated content may be incorrect.

This DNN achieved strong performance, similar to the top scikit-learn models, demonstrating the potential of deep learning for this task when appropriate computational resources are available. TensorBoard was used to monitor the training process and visualize model metrics.

**Model Interpretation with SHAP Values**

SHAP (SHapley Additive exPlanations) values were used to interpret the model predictions, identifying the contribution of each gene to the classification outcome.

**Binary Classifier SHAP:** For binary classification (Control vs. DCM), the SHAP summary plot (shap\_summary\_plot\_binary\_gene\_names.png) was generated from the **TensorFlow DNN model run on Google Colab**. This plot illustrates key genes like HBA2 and HBA1 (high expression associated with DCM) and SERPINA3, IL1RL1, SFRP4 (high expression associated with Control) as drivers of the model's predictions.

A screen shot of a computer

AI-generated content may be incorrect.

**Multi-Class Classifier SHAP:** For the 3-class problem, SHAP values were calculated for the **SVM model**. The KernelExplainer was used, returning SHAP values as a 3D array (samples, features, classes). To visualize, three separate summary plots were generated:

A screen shot of a computer screen

AI-generated content may be incorrect.A screen shot of a computer

AI-generated content may be incorrect.A screen shot of a computer screen

AI-generated content may be incorrect.

These plots show the genes most influential for predicting each specific class. For instance, a gene might strongly push towards a 'DCM' prediction, while another might strongly push away from 'HCM'.

**Web Application Development**

An interactive web application (app.py) was developed using Streamlit to wrap the entire analysis pipeline. The application allows users to:

* Select between binary and multi-class classification.
* Choose from various scikit-learn models.
* View dynamic PCA plots, model evaluation metrics (classification reports, confusion matrices, ROC AUC scores).
* Trigger on-demand SHAP value calculation and visualize feature importance for the selected model and classification type.

**Conclusion & Discussion**

This project successfully developed a comprehensive pipeline for heart disease classification from RNA-Seq data. The binary classification proved highly accurate, while the multi-class problem, particularly the HCM class, presented a more challenging scenario due to sample imbalance. Feature selection using the top 1000 most variable genes effectively balanced performance and computational efficiency. SHAP values provided crucial insights into the biological drivers of model predictions.

**Limitations:** The primary limitation is the small sample size for the HCM group, impacting multi-class performance. The reliance on CPU for KernelExplainer and TensorFlow's incompatibility with the local environment also posed challenges. The current web app uses pre-loaded data; user data upload would be a valuable future enhancement.

**Future Directions:** Future work could involve collecting more data for underrepresented classes, exploring more advanced feature selection techniques, implementing more sophisticated multi-class deep learning models (e.g., on cloud GPUs), and integrating additional biological pathway analysis.

**Data and Code Availability**

* **Dataset:** Magnetique dataset available on Zenodo: <https://zenodo.org/record/7148045>
* **Code Repository:** The project code is available on GitHub: <https://github.com/xiyasong/DDLS_Project.git>
* The web server is accessible on: <https://xiyasong-ddls-project-app-k830az.streamlit.app/>

**Acknowledgments**

This project was developed with significant assistance from the Gemini-CLI AI agent, which facilitated code generation, debugging, and project management. Its contributions are detailed in the AI deep research log.

**References**

Britto-Borges, T., Ludt, A., Boileau, E. *et al.* Magnetique: an interactive web application to explore transcriptome signatures of heart failure. *J Transl Med* **20**, 513 (2022). https://doi.org/10.1186/s12967-022-03694-z

**Appendices**

* **AI Deep Research Log: final\_history.json**