**Brief Overview of the Project:**

In this project, we aimed to develop a machine learning model capable of predicting the mechanisms of action (MoA) for various compounds based on their gene expression and cell viability profiles. The dataset used for this project was obtained from the LISH-MoA competition on Kaggle, which contains information on over 23,000 compounds and their effects on various biological targets. We employed four different machine learning algorithms, namely Logistic Regression, Support Vector Machines (SVM), Random Forest, and Neural Networks to build our predictive models. The performance of these models was evaluated using the log loss metric, which measures the accuracy of probabilistic predictions.

**1.2 Importance of MoA Prediction in Drug Discovery**

Mechanisms of action prediction plays a crucial role in the drug discovery process, as understanding how a compound interacts with biological targets can provide valuable insights into its potential therapeutic effects and side effects. Accurate MoA predictions can facilitate the selection of promising drug candidates, potentially reducing the time and costs associated with drug development. Moreover, knowledge of a compound's MoA can also aid in the identification of new therapeutic applications for existing drugs, a process known as drug repurposing.

**1.3 Outline of the Report**

The remainder of this report is organized as follows:

Section 2 provides a detailed description of the dataset used in this project.

Section 3 explains the machine learning algorithms employed, including their background information, equations, and figures.

Section 4 describes the experimental setup, including data preprocessing, model implementation, and performance evaluation.

Section 5 presents the results of the experiments, comparing the performance of the different models, and interpreting the findings.

Section 6 summarizes the main conclusions drawn from the project and suggests potential improvements for future work.

Section 7 lists the references cited throughout the report.

Section 8 includes an appendix containing documented computer code listings for the project.

**Description of the Dataset**

**2.1 Source: LISH-MoA competition on Kaggle**

The dataset used in this project was obtained from the LISH-MoA competition on Kaggle. This dataset contains gene expression and cell viability data for over 23,000 compounds, along with their mechanisms of action (MoA) annotations.

**2.2 Description of the Data Files**

The dataset is divided into the following data files:

train\_features.csv: Contains the gene expression and cell viability data for the training set. It includes 23,314 samples with 876 columns, where the first column represents the unique compound identifier ('sig\_id') and the remaining columns correspond to the gene expression and cell viability features.

train\_target\_scored.csv: Contains the binary MoA annotations for the training set, with one row per compound and one column per MoA.

**2.3 Explanation of the Features**

The features in the dataset can be divided into two categories: gene expression and cell viability.

Gene expression features (g-0 to g-771): These features represent the expression levels of various genes after exposure to the compounds. Higher or lower expression levels indicate the activation or inhibition of specific genes, which can help infer the compound's mechanism of action.

Cell viability features (c-0 to c-99): These features represent the viability of different cell lines after exposure to the compounds. Cell viability is a measure of cell health and can provide insights into the compound's cytotoxicity or potential side effects.

**Data Exploration and Visualization**

We investigated the dataset's characteristics using various visualizations. This allowed us to better understand the distribution of features, differences in cell viability between treated and control samples, the impact of treatment time on cell viability, and the correlation between gene expression and cell viability features.

1. We plotted histograms for selected gene expression (g) and cell viability (c) features to observe their distributions. This helped us identify any potential outliers, skewness, or other patterns in the data.

Chart, histogram

Description automatically generated

The range of gene expression values is quite wide, spanning from significantly negative to positive values. This demonstrates that the dataset contains genes with varying degrees of activation or inhibition. The gene expression values seem to have a roughly symmetrical distribution around zero, suggesting that many genes have balanced expression levels, with some being upregulated (positive values) and others being downregulated (negative values).

Chart, histogram

Description automatically generated

The distributions of the selected c- features appear to be approximately normal (bell-shaped) with some slight deviations. This suggests that cell viabilities in these features follow a common pattern, and extreme values are less frequent. The histograms show that the peak (mode) for each feature is around 0, which could be a result of the data normalization process applied to the dataset. There is a noticeable range in the x-axis values for these features, spanning from around -10 to 6, high negative values indicate a high number of dead cells, while high positive values indicate a high number of living cells

1. We compared the mean cell viability between treated and control samples. This comparison allowed us to identify any significant differences in cell viability due to the presence or absence of treatment.

Chart, line chart, histogram

Description automatically generated

The above KDE plot helps visualize the impact of treatments on cell viability for a specific feature (c-30) by comparing it to control samples. The differences in the distributions provide insights into how various treatments may affect cell viability across samples. The treated samples exhibit a broader and slightly more evenly distributed range of cell viability values compared to the control samples. This indicates that the treatment has a more diverse effect on cell viability in various samples. The control samples show a sharp peak around 0, suggesting that in the absence of treatment, cell viability tends to remain stable and less variable. This suggests that some treatments might have a significant impact on cell viability, causing either an increase in the number of living cells or a decrease leading to cell death.

1. We analyzed the impact of treatment time on cell viability by comparing the mean cell viability at different treatment times (24, 48, and 72 hours). This helped us to determine if treatment duration has any effect on cell viability.

Chart, line chart

Description automatically generated

The distribution of cell viability values varies across different treatment durations. This suggests that treatment time has a noticeable effect on cell viability for the selected feature. The plot illustrating the impact of treatment time on cell viability for a specific feature reveals that treatment duration has a considerable influence on cell viability. The observed variability in cell viability values across different treatment times emphasizes the need to account for treatment duration when analyzing the effects of treatments and developing predictive models.

1. We created correlation matrices for gene expression and cell viability features to explore the relationships between these features. This allowed us to identify any strong correlations between features, which could be further investigated for potential feature selection or dimensionality reduction.

Chart

Description automatically generated with low confidenceThe heatmap plot of the correlation between cell viability features in treated samples reveals a wide range of relationships, from strong positive to negative correlations. These correlations provide valuable insights into how cell viability features interact and respond to various treatments, which can help guide future analysis and model development.

A picture containing diagram

Description automatically generated

By exploring and visualizing the data, we gained valuable insights into the dataset's structure and relationships, which can inform our future modelling sdecisions and improve our understanding of the underlying biological processes.

**2.4 Data Pre-processing Steps**

Before training the machine learning models, the following preprocessing steps were performed on the dataset:

Merging the data: The train\_features and train\_target\_scored datasets were merged on the "sig\_id" column.

Separating categorical columns: The categorical columns 'cp\_type', 'cp\_time', and 'cp\_dose' were separated from the numerical columns for preprocessing.

Standardizing the numerical data: The gene expression data was standardized using the RobustScaler. This ensures that all gene expression features are on a similar scale and prevents any one feature from dominating the model's predictions.

Dimensionality reduction using PCA: Principal Component Analysis (PCA) was applied to the gene expression data, reducing the dimensionality while retaining 95% of the variance in the data. This step helps visualize the dataset and potentially speeds up the machine learning algorithms.

Merging the preprocessed data: The scaled gene expression data, cell viability data, and one-hot encoded categorical data were merged into a single dataset.

Splitting the data: The data was split into training and validation sets using an iterative stratification approach with an 80-20 ratio, which ensures that the distribution of MoA annotations in both sets is similar.

Scaling and dimensionality reduction on the entire dataset: The merged dataset was then scaled using the RobustScaler, and PCA was applied to reduce dimensionality while retaining 95% of the variance in the data.

**3. Machine Learning Algorithms**

In this study, three machine learning algorithms were employed to predict the mechanism of action (MoA) of the compounds: Logistic Regression, Support Vector Machines (SVM), and Random Forest.

**3.1 Logistic Regression**

Logistic Regression is a linear statistical model used for binary classification tasks. It is an extension of the linear regression model for classification purposes. The logistic function, also known as the sigmoid function, is applied to transform the output into probabilities that sum to 1. These probabilities can then be used to predict the class labels.

**3.2 Support Vector Machines (SVM)**

Support Vector Machines (SVM) is a powerful and flexible classification algorithm that can be used for both linear and nonlinear classification tasks. SVM aims to find the hyperplane that best separates the classes in the feature space. The algorithm maximizes the margin, which is the distance between the hyperplane and the closest data points, called support vectors, from each class.

**3.3 Random Forest**

Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. The algorithm creates multiple trees, each trained on a random subset of the training data and averages their predictions. This approach reduces overfitting and improves the model's generalization performance.

**3.4 Model Training and Evaluation**

We have demonstrated the process of training and evaluating the three machine learning algorithms. The dataset is first split into training and validation sets using an iterative stratification approach. Then, the MultiOutputClassifier is applied to each base model to handle the multi-label classification problem. Each model is trained using the fit method, and the predictions are made using the predict\_proba method if available or the predict method otherwise. Finally, the log loss is calculated for each model to evaluate its performance.

**4.Experimental Setup**

The experimental setup involves several key steps, including data splitting, feature scaling, dimensionality reduction, performance evaluation, and code organization.

**4.1 Data Splitting**

The dataset is split into training and validation sets using an iterative stratification approach, which helps in maintaining the class distribution across the splits. The iterative\_train\_test\_split function from the skmultilearn.model\_selection library is employed for this purpose, with 80% of the data used for training and 20% for validation.

**4.2 Feature Scaling**

Feature scaling is performed using the RobustScaler from the sklearn.preprocessing library. This method is less sensitive to outliers, as it scales the features using the interquartile range (IQR) instead of the mean and standard deviation. The RobustScaler is fit on the training data and then applied to both the training and validation sets.

**4.3 Dimensionality Reduction**

Dimensionality reduction is performed using Principal Component Analysis (PCA) from the sklearn.decomposition library. PCA is a technique that transforms the data into a new coordinate system with orthogonal axes, which helps in reducing the number of features while retaining most of the variance in the data. In this study, PCA is configured to retain 95% of the explained variance.

**4.4 Performance Metric**

The performance of the machine learning models is evaluated using the log loss metric. Log loss, also known as logarithmic loss or cross-entropy loss, measures the performance of a classification model where the predicted output is a probability value between 0 and 1. The log loss for each model is calculated using the log\_loss function from the sklearn.metrics library.

**4.5 Code Organization**

The code is organized into subroutines and functions for modularity and ease of understanding. The functions in the provided code snippet include:

plot\_individual\_histograms: Plots histograms for selected features in the dataset.

plot\_cell\_viability\_difference: Visualizes the difference in cell viability between treated and control samples for a given feature.

correlation\_matrix: Computes and displays the correlation matrix for the selected features in the dataset.

**5 Results:**

The performance of the three machine learning models (Logistic Regression, SVM, and Random Forest) is evaluated on the validation set using the log loss metric. The results are compared across different models to identify the best-performing model.

**5.1 Performance of Each Model on the Validation Set**

The log loss values for each model on the validation set are as follows:

Logistic Regression: [log\_loss value]

SVM: [log\_loss value]

Random Forest: [log\_loss value]

**5.2 Comparison of the Results Across Different Models**

A comparison of the log loss values indicates that the [best\_model\_name] model performs the best on the validation set, with the lowest log loss value among the three models.

**5.3 Interpretation of the Results**

The log loss metric is used to evaluate the performance of the classification models. Lower log loss values signify better model performance. The comparison of the results suggests that the [best\_model\_name] model provides the most accurate predictions among the tested models for this specific dataset.

**5.4 Tables and Figures to Support the Results**

|  |  |
| --- | --- |
| Model | Log Loss |
| Logistic Regression |  |
| SVM |  |
| Random Forest |  |

The table clearly shows the log loss values for each model, making it easy to compare their performance on the validation set. The [best\_model\_name] model has the lowest log loss value, indicating better performance compared to the other two models.

**6 Summary and Conclusions**

**6.1 Recap of the Main Findings**

This study aimed to predict cellular responses to different drugs using three machine learning models: Logistic Regression, SVM, and Random Forest. The dataset was pre-processed, and dimensionality reduction was applied using PCA. The performance of each model was assessed using the log loss metric on the validation set. The main findings of the study are:

The [best\_model\_name] model exhibited the best performance among the three models, with the lowest log loss value.

**6.2 Lessons Learned from the Project**

The project demonstrated the importance of data preprocessing, feature scaling, dimensionality reduction, and model evaluation in building and comparing machine learning models. It also showed that different models may perform differently on the same dataset, underscoring the need to try multiple models to identify the best-performing one.

**6.3 Suggestions for Future Improvements**

There are several potential avenues for improving the predictive performance of the models:

Better feature engineering: Investigating and creating more informative features could lead to improved model performance. Trying more advanced models: Exploring deep learning models or other advanced machine learning techniques could yield better results. Exploring ensemble techniques: Combining the predictions of multiple models can often lead to more accurate predictions.

**7 References**

[List of sources cited in the report]

**8 Appendix**

**8.1 Documented Computer Code Listings**

[Include the documented code listings used in the project]

**8.2 Description of the Functions and Their Usage**

[Provide a brief description of the functions used in the code and explain their usage]