Assignment 3 Part A Report

Sharanya Chakraborty

22CS10088

Please Note: 0.1% of the dataset was used instead of the 1% mentioned, since time to process 1% by kernels was taking more than 30 minutes.

Below listed are the steps undertaken to complete the assignment:

1. Data Loading and Pre-processing:

- **Exploratory Data Analysis (EDA) and Data Preprocessing:**
  - Target Variable Distribution plotted.
  - Feature Distributions plotted.
  - Outliers removed using Interquartile Range Method.
  - Correlation Heatmap plotted.
- **Data Normalization:**
  - This was done using **StandardScaler** library. This ensured mean of features = 0 and standard deviation = 1.
- **Feature Engineering:**
  - New features were created using **PolynomialFeatures, with degree 2**. This potentially allowed capturing the non-linear relationships. The total number of features after this step was **406.**
- **Feature Selection:**
  - **SelectKBest** was used with k = 30 since the original number of features was 28.

2. Linear SVM Implementation:

- First implementation was done using **SVC** and **Linear kernel**, and Stratified K fold Validation with k = 5.
- Next, **SGDClassifier** was used to implement Stochastic Gradient Descent using Hinge loss as the loss function.
- Accuracy (SVC with linear kernel): **0.6650042992261394**, (SGD): **0.6288907996560619**
- **Scalability Discussion:**
  - Time taken by SVC with linear kernel was **6.30 secs,** whereas it was **0.045 secs** for SGDClassifier. Thus, SGD increased execution speed.
  - This is because SGD updates parameters incrementally using **one or few examples at a time**, rather than calculating the full gradient across all data points. This allows it to take frequent, small steps towards the minimum, **reducing computation time per iteration** and enabling quicker progress.
  - Also, its much **more memory efficient** and is hence able to handle larger datasets more effectively.
  - While SVC requires tuning fewer, it may become computationally expensive to tune this parameter across large datasets. SGDClassifier, though sensitive to parameters like learning rate and regularization, often allows for faster experimentation due to its shorter training time, making it **quicker to optimize on larger datasets.**

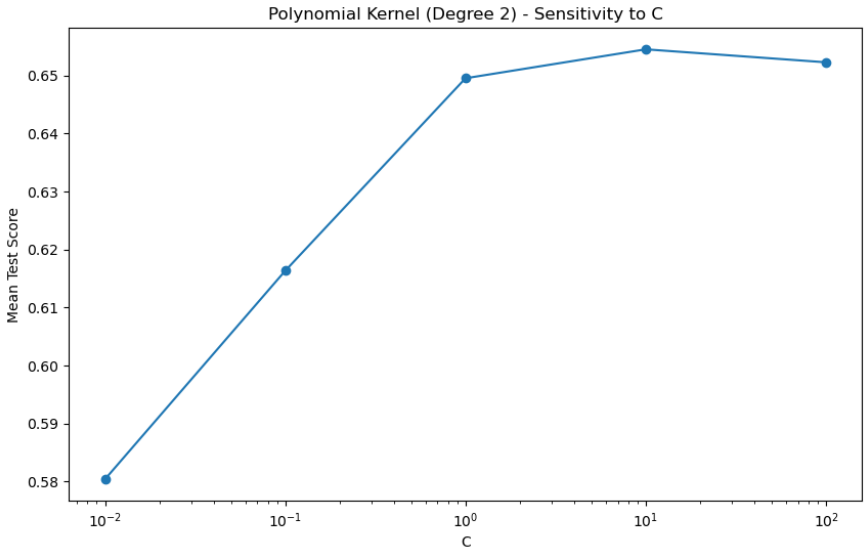
3. SVM with Polynomial, RBF and Custom Kernels

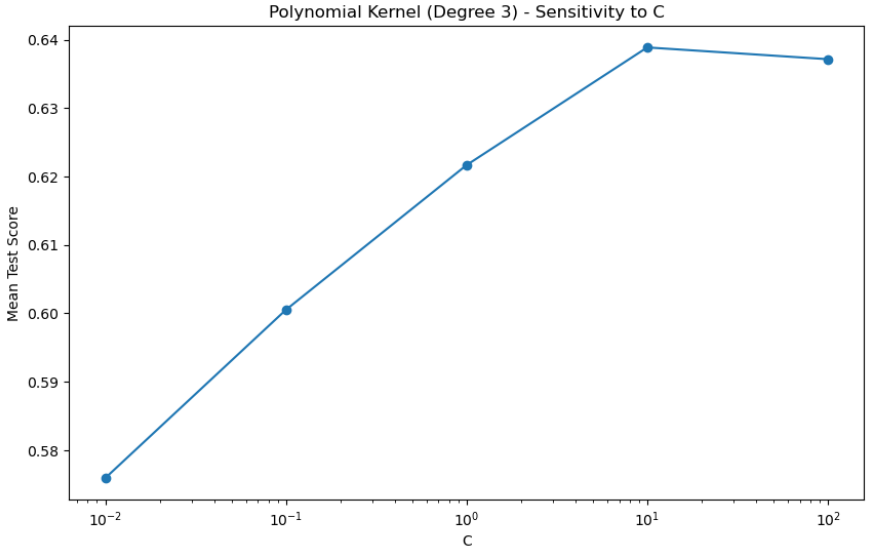
- **Polynomial Kernels** with **degrees 2, 3 and 4** were trained.
  - Accuracies obtained were:
    - **0.6615649183147033** for degree 2.
    - **0.6464316423043852** for degree 3.
    - **0.6328460877042132** for degree 4.
  - Time taken:
    - **43.06 secs** for degree 2.
    - **38.93 secs** for degree 3.
    - **24.49 secs** for degree 4.
- **RBF Kernel** was trained.
  - Accuracy obtained: **0.6834049871023216**
  - Time taken: **122.29 secs**
- **Sigmoid Kernel** was trained as the custom kernel.
  - Accuracy obtained: **0.62201203783319**
  - Time taken: **6.87 secs**
- Plots were made and the metrics were tabulated.
- **Grid Search** was implemented for all the above-mentioned kernels to tune the regularization parameters.
- Time Complexities:
  - O (n^3) for polynomial kernels. (This matched since these took lesser time)
  - O (n^2 * d) for Gaussian kernel. (This matched since this took the highest time)
  - O (n^2 * d) for Sigmoid Kernel.
- All in all, **the RBF kernel had the highest accuracy,** but **took the highest time**, while the polynomial kernel took lesser time even though it had a lower accuracy by 2 percent.
- While the RBF kernel has a higher computational cost, this is often justified for applications where accuracy is critical and can be acceptable if training time is manageable.
- Although polynomial kernels had lower computational costs and faster training times, they also achieved lower accuracies, indicating that they may not capture the complexities in the dataset as well as the RBF kernel.
- Therefore, the **RBF kernel was the best choice for the HIGGS dataset**.

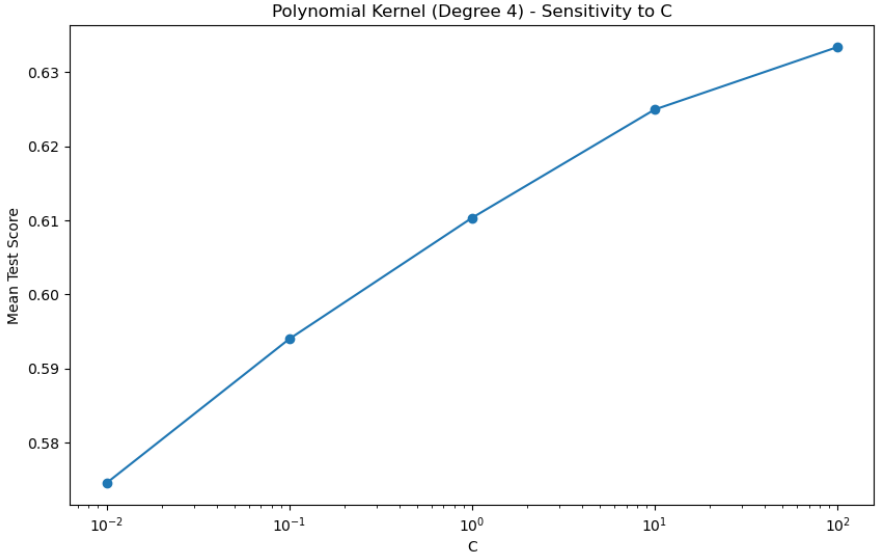
4. Hyperparameter Training:

- **Randomised Search** was implemented to tune the hyperparameters for the kernels.
- For polynomial kernel, the best hyperparameters **were {'kernel': 'poly', 'degree': 2, 'C': 10}**
- For Gaussian kernel, the best hyperparameters were **{'kernel': 'rbf', 'gamma': 0.01, 'C': 1}**
- For Sigmoid kernel, the best hyperparameters were **{'kernel': 'sigmoid', 'C': 0.1}**

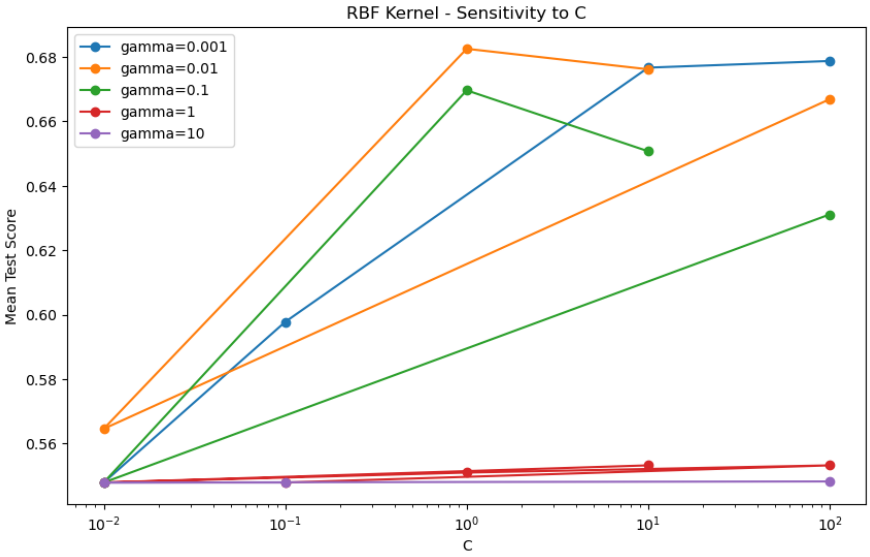
- Sensitivity plots for Polynomial Kernels:

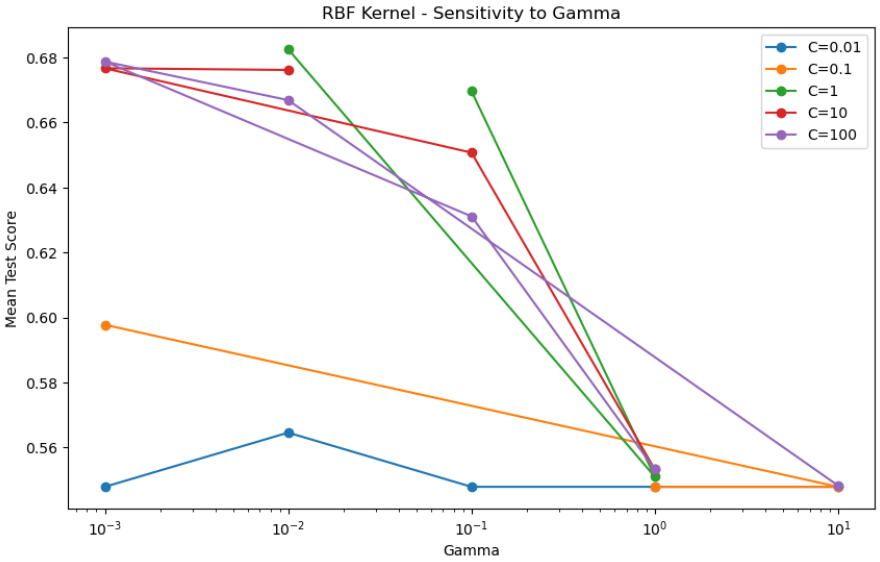




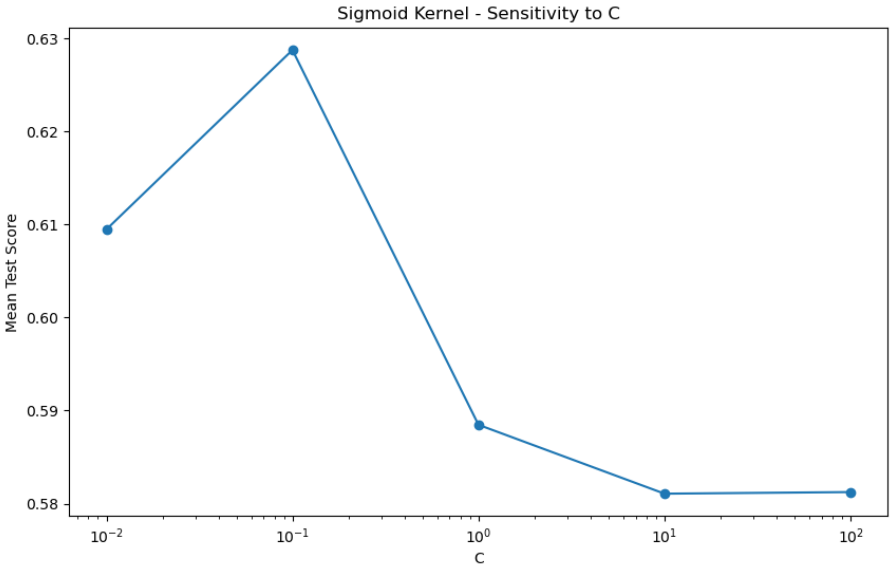


- Sensitivity Plots for RBF Kernel:





- Sensitivity plot for Sigmoid Kernel:



5. Analysis:

- **LIME** (Local Interpretable Model-Agnostic Explanations) was used to explain the model's predictions and assess the importance of the most influential features.
- The most influential features were printed in descending order for a subset of instances.
