Don’t Overfit!

Hardik Gourisaria
PES1201700129

Mayank Agarwal
PES1201701349

Amit Kumar
PES1201701295

***Abstract*—This work introduces a Classification Model which when trained on a dataset of 300 features and 250 training samples, does not overfit the data set and provides an AUCROC score of 0.848 on a scale of 0 to 1 for the testing dataset that has around 20,000 unlabeled testing samples. This work comprises two main aspects. They are Feature Engineering and Classification.**

***Index Terms*—Feature Engineering, Overfitting, Dimensionality Reduction, Classification**

I. Introduction

THIS work aims to design a model that relies on the concepts of feature engineering and classification to fit a model to a dataset with 300 features and 250 training samples such that the model does not overfit [3] the data and provides high accuracy results for 19750 unlabeled testing samples.

The problem is picked up from a Kaggle Competition called “Don’t Overfit!” [1] and hence the title.

Two main aspects to be considered for solving the problem are Feature Engineering [5] and Classification. Feature Engineering is required so as to reduce the number of features in the dataset, thus reducing overfitting. It also helps us consider strictly, only the features that are related to the desired output. This is an essential step and the reason has been discussed in the following sections. Classification is required because the problem is essentially a classification problem where we have to predict the class levels of a sample based on the feature values provided.

II. Feature Engineering

A. What is it?

Feature engineering [5] is the process of using domain knowledge to construct a dataset with features pertaining to the required problem statement. Performing feature engineering makes the Machine Learning models work well. If features are not picked correctly, a model trained on the dataset may not yield desirable results due to redundancies, overfitting or irrelevant data. The basic essence is to make life easier for the machine learning model developer as picking the right and relevant features simplifies the understanding phase, development phase and deployment phase of the model. This provides better results for the addressed problem with shorter development time and at a lesser cost.

B. Techniques Involved

Feature engineering can be used to address many problems in the dataset such as fixing faulty data, replacing or removing missing values, dimensionality reduction [2], etc. In this work, dimensionality reduction is by far the most important aspect of feature engineering that is essential to develop the machine learning model that performs well. Given the training and testing datasets have 300 features, it is necessary to reduce the number of features so as to reduce the model complexity, improve understandability, reduce development time and reduce the chances of overfitting etc. to name a few. It has been explained briefly in the following sections.

III. Overfitting

A. What is it?

Overfitting [3] is a situation where the model provides high accuracy on the training dataset, however fails to do the same on the testing dataset. It can also be said that the model learns the training data too well and is unable to generalize well on the testing dataset. An overfitted model is a statistical model that contains more parameters than can be justified by the data. In the case of our work, the model has a high chance of overfitting as there are 300 features that the model can be trained on. This leads to the increase in complexity of the model and hence overfitting.

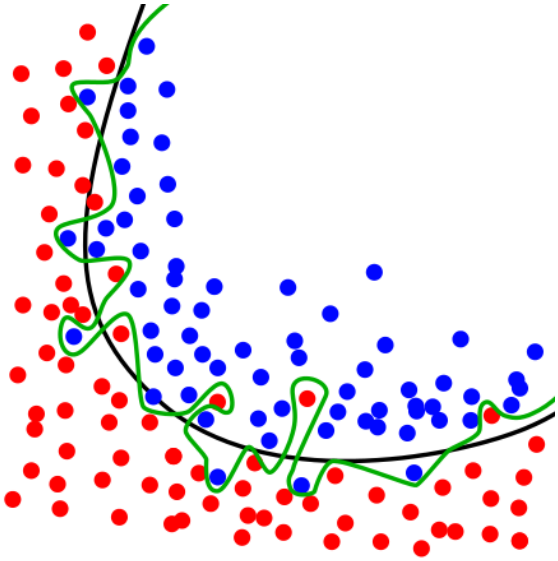

Fig 1. Diagram comparing a overfitted model against a well fitted model. The green line represents the overfitted model and the black line represents the well fitted model.

It can also be said that when the model is trained on many features, it starts to learn the noise in the data too, and hence does not provide correct predictions because it focusses too much on the moisy details in the data.

B. Reducing Overfitting

Some of the common ways to reduce overfitting [4] are to use methodologies like cross validation, early stopping, pruning, regularization, dimensionality reduction etc. just to name a few. In our work the main focus is on using dimensionality reduction to reduce the number of parameters in the model and hence reducing the complexity and chance of overfitting.

IV. Dimensionality Reduction

A. What is it?

In machine learning problems, there are often too many variables called features based on which the prediction is performed. The higher the number of features, the harder it gets to visualize the training dataset and come up with appropriate algorithms to solve the problem. Sometimes, the features are highly correlated, and hence redundant. This is where dimensionality reduction algorithms come into picture. Dimensionality reduction reduces the number of features under consideration, by picking only the relevant features or dropping the redundant features.

B. Feature Selection

Here, we try to obtain a subset of the original feature set from the dataset, to model the problem. It usually involves three ways:

1. Filter
2. Wrapper
3. Embedded

C. Feature Extraction

This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser number of dimensions.

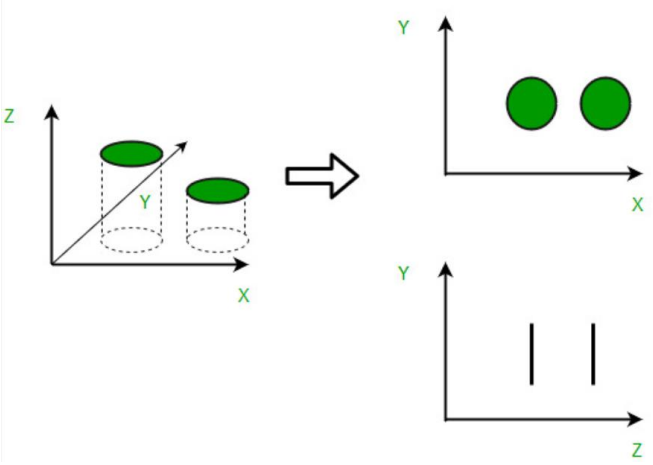

Fig 2. Diagram depicting dimensionality reduction by feature extraction. The 3D model is reduced to a 2D model which is easier to understand by the removal of one axis or feature

D. Methods

The various methods used for dimensionality reduction include:

1. Principal Component Analysis (PCA)
2. Singular Value Decomposition (SVD)
3. Lasso Regression

Dimensionality reduction may be both linear or non-linear, depending upon the method used.

E. Advantages

- It helps in data compression, and hence reduced storage space.
- It reduces training and testing time and saves cost.
- It also helps remove redundant features, if any and hence reduces chances of overfitting hence improving the model performance.

F. Disadvantages of Dimensionality Reduction

- Important details may be lost.
- In the case of PCA, it finds linear correlations between features which is undesirable sometimes.
- How many principle components are to be considered may not be easy to determine and some basic thumb rules may need to be followed.

V. Exploratory Data Analysis

Exploratory Data Analysis was performed on the dataset and the following inferences were made:

1. The data was found to be clean with no missing values.
2. The dataset is used for a binary classification problem and following are the statistics of the target values.

count	250.000000
mean	0.640000
std	0.480963
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

3. It can be inferred from above that the class level with value 1 has higher frequency compared to class level with value 0.
4. A logistic regression model was constructed and used for preliminary analysis and prediction on the dataset. The model achieved a 100% accuracy on the training dataset while only a AUCROC [7] score of 0.662 on the test set when evaluated on Kaggle. This clearly indicates that the model has overfitted the data.
5. Extra Tree Classifier was used for selecting significant features from the dataset and 25 features were selected. A heat map of the features was plotted and it was found that the features are not highly correlated.
6. It was also observed that the features are normally distributed and unimodal.

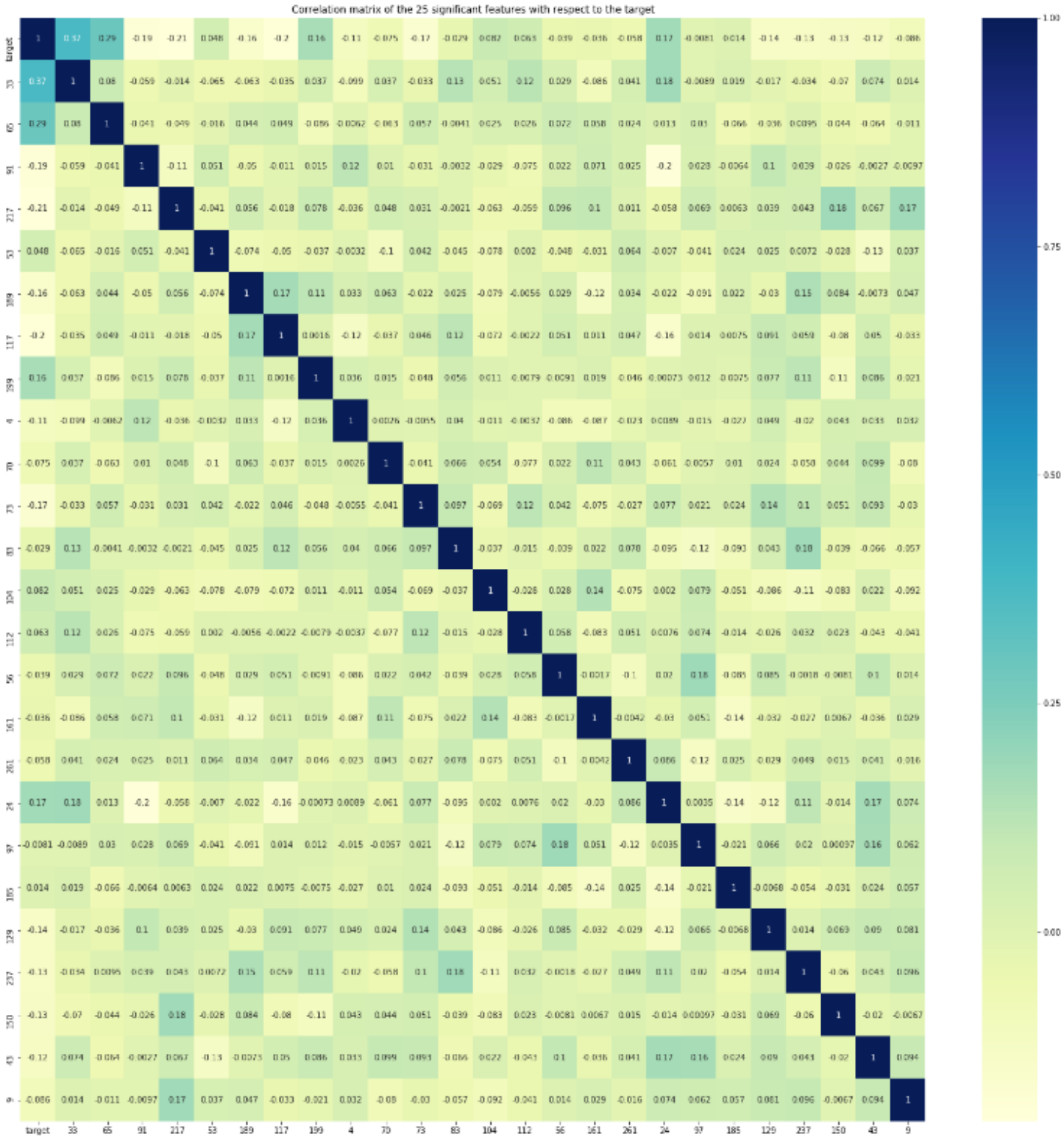

Fig 3. Heap map of the significant features selected with respect to the target vector. As can be observed from the heat map, the significant features are weakly correlated.

7. Since the origin of the data is unknown to us, there is a possibility that the data comes from a time series. This possibility cannot be ruled out as it can highly impact the output labels of out model and hence need to be considered.
8. ACF and PACF plots of the data show that data being considered is not a time series data and hence the notion of the data origination from a time series can be safely dropped.

VI. Selecting a Classification Model

Selecting a classification model that work well is of utmost importance as the end output that we expect from out machine learning model for the addressed problem statement are the class levels of the input features. Several Classification models were implemented and their accuracies were computed using the Kaggle evaluation platform. These are some of the Classification Models that were attempted without any dimensionality reduction and their results compared.

A. Logistic Regression Classifier

Logistic Regression was used because it is very easy to implement and servers as a performance baseline for many other algorithms. It also performs really well in most applications. Most important thing to note about Logistic Regression is that it is highly interpretable and does not require high computation resources, just to name a few. It is also very well suitable for binary classification problems as in the case of our problem statement.

B. Support Vector Classifier

Support Vector Classifier was tried out as it is very effective in high dimensional spaces where the number of dimensions is greater than the number of samples [8]. Support vector classifiers are also considered to be very memory efficient.

C. Gaussian Naïve Bayes Classifier

Gaussian Naïve Bayes Classifier was tried out as it is very easy to implement and not computationally expensive. It can achieve very good results in cases where the training data size [9] is very small as the probability of overfitting is lesser when compared to other classification models.

D. AdaBoost Classifier

AdaBoost Classifier was implemented as the implementation is simple and the algorithm is extremely fast. It also has the flexibility to combine with any machine learning algorithm and there are not many parameters that need to be tuned [10]. It works really well for binary as well as multi class problems.

E. Artificial Neural Network

A simple Artificial Neural Network Classifier was designed and implemented. Motivation behind trying out an Artificial Neural Network was that they are extremely tolerant to noisy or incomplete data. Artificial Neural Networks also have the capability of recognizing complex patterns in the data when can be used to distinctly classify the training samples. However, a major disadvantage of the Artificial Neural Networks is the large amount of initial training time and the difficulty to tune the parameters.

Table 1 below lists and compares the result obtained after training our model on the data and evaluating the out on the Kaggle evaluation platform.

Classification Model	**AUCROC Score on Test Data**
Logistic Regression Classifier	0.662
Support Vector Classifier	0.663
Artificial Neural Network Classifier	0.660
Gaussian Naïve Bayes Classifier	0.568
AdaBoost Classifier	0.542
Decision Trees Classifier	0.568
Gaussian Process Classifier	0.526
Random Forest Classifier	0.542
K Nearest Neighbors Classifier	0.560

Table 1. Comparison of various Classification Models implemented and tested on the dataset without any dimensionality reduction.

As Evident from the above table, Logistic Regression Classifier, Support Vector Classifier and Artificial Neural Network Classifier perform the best on our dataset. On further analysis, we can observe that there is not much difference in the accuracy of the result produced by the above models and hence taking into account the implementation simplicity, understandability and implementation cost and time, we moved ahead after selecting Logistic Regression as our Classification Model.

Logistic Regression can be Defined as follows:
Let β be the parameters of the model and x_i be the features of training sample in the model. Then the probability of the training samples can be given by the following equation.

$$p_i = \frac{1}{1 + e^{-\beta^T \mathbf{x}_i}}$$

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))]$$

Given above is the Cost function of Logistic Regression where h_θ is the predicted value for the training sample in the particular iteration.

VI. Selecting a Dimensionality Reduction Technique

A few of the dimensionality reduction techniques were tried out and results were compared. The dimensionality reduction techniques tried out are:

1. Principal Component Analysis
2. Singular Value Decomposition
3. Lasso Regression

Given below are the AUCROC score obtained after reducing dimensions of the data using the mentioned techniques and training a Logistic Regression model on the reduced data.

Dimensionality Reduction Technique	AUCROC Score on Test Data
Lasso Regression	0.848
Singular Value Decomposition	0.724
Principle Component Analysis	0.649

It was observed that Lasso regression when coupled with Logistic Regression yielded very AUCROC value of 0.848 on the testing data when tested on Kaggle.

Lasso Regression is technically a regularization technique where the penalty term is the mean absolute error rather than mean squared error. It can also we called L1 regularization. The advantage of Lasso regression is that it tends to set the coefficients of the redundant features to 0 and hence acts as a dimensionality reduction technique as setting the coefficient value of a feature to 0 removes any contribution of the feature towards fitting the model and obtain the target labels.

The optimization function for L1 regularized Logistic Regression is as follows:

$$\max_{\beta} log \prod_{i}^{n} \binom{m_i}{y_i} p_i^{y_i} (1-p_i)^{m_i - y_i} - \lambda ||\beta||_1$$

where m_i is the weight of the training sample and y_i is the class label of the training sample. λ is the coefficients of each parameter.

After performing Lasso Regularization, the coefficients of the features were analyzed and it was found that only 124 of the initial 300 features were of significance. The number of features was reduced to about 40% of the initial number of features. Due to this, the model performed much better after applying Lasso regularization than before applying Lasso Regularization. The feature names have no significance and have not been mentioned as such, but it is the feature values that are highly significant to us. The AUCROC score on the test data increased from 0.662 to 0848 which is a considerable rise. With this score, we can be placed at 311^{th} position on the Kaggle Competition Leaderboards.

Many other models were tried out and the accuracies were measured. However, none of the other models produced an accuracy coming even close to the accuracy yielded by the Logistic regression with Lasso Regularization model. Hence these model were scrapped.

VII. Result and Conclusion

After comparing multiple models Logistic Regression with Lasso Regularization was found to yield the best AUCROC score of 0.848 when tested on the Kaggle evaluation platform. We believe this to be an impressive result given that the origin of the dataset is unknown and the number of features in the dataset far exceed the number of training sampled in the dataset.

VIII. Work Done and Next Steps

Dimensionality Reduction on the dataset was performed using Lasso Regression Regularization on a Logistic Regression Classifier. The Classifier was then trained was trained and tested on the test dataset. The results when tested on the Kaggle evaluation platform, produced an AUCROC score of 0.848. This clearly states that the model which was initially overfitting without Lasso Regression Regularization, does not do so anymore. Future work can involve, further working on the feature engineering aspect of the model by attempting to implement more sophisticated Dimensionality Reduction Techniques.

References

[1] https://www.kaggle.com/c/dont-overfit-ii/overview
[2] https://www.geeksforgeeks.org/dimensionality-reduction/
[3] https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/
[4] https://en.wikipedia.org/wiki/Overfitting#Remedy
[5] https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114
[6] https://www.theanalysisfactor.com/the-distribution-of-independent-variables-in-regression-models/#targetText=There%20are%20NO%20assumptions%20in,continuous%20or%20discrete)%20independent%20variables.&targetText=They%20do%20not%20need%20to%20be%20normally%20distributed%20or%20continuous.
[7] https://en.wikipedia.org/wiki/Receiver_operating_characteristic
[8] https://medium.com/@dhiraj8899/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107
[9] https://towardsdatascience.com/the-naive-bayes-classifier-e92ea9f47523
[10] https://www.educba.com/adaboost-algorithm/
[11] http://web.stanford.edu/~hastie/Papers/glmnet.pdf
[12] Class Lecture Presentations

Individual Contributions

A. Hardik Gourisaria (PES1201700129):

Contributed to selecting the Classification models, little bit of selecting the Dimensionality Reduction technique and report writing. Cleaned code for other parts of the project.

B. Mayank Agarwal (PES1201701349):

Contributed to Exploratory Data Analysis, selecting the Dimensionality Reduction technique and report writing. Controlled the work flow of the project.

C. Amit Kumar (PES1201701295):

Contributed majorly to the Exploratory Data Analysis, Literature Survey and selecting the Dimensionality Reduction technique.