SANTANDER CUSTOMER TRANSACTION PREDICTION

TO EVALUATE DIFFERENT MODELS FOR BINARY-CLASSIFICATION

AJEYA KEMPEGOWDA
DEEPANSHU PARIHAR
HARISH RAMANI
I YEAR MSDS

PROBLEM STATEMENT

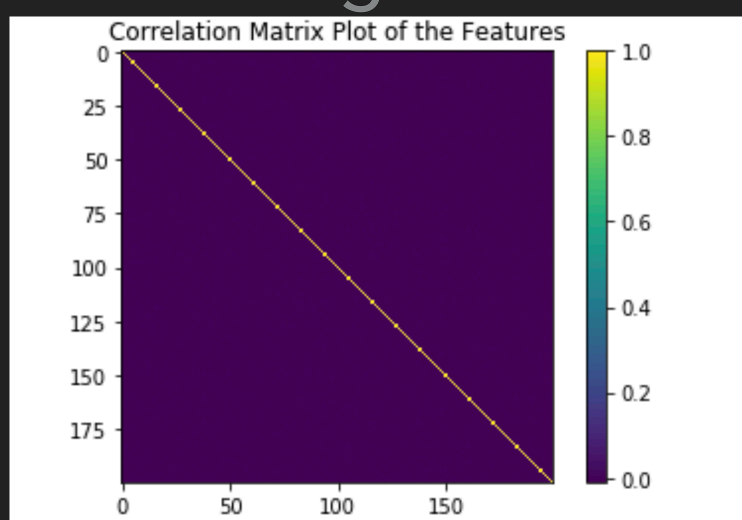
- To predict future customer transactions using Santander bank's transaction data.
- To build ML models that tackle binary-classification problem.
- To explore new family of Neural networks(Neural ODE) and to evaluate the results.

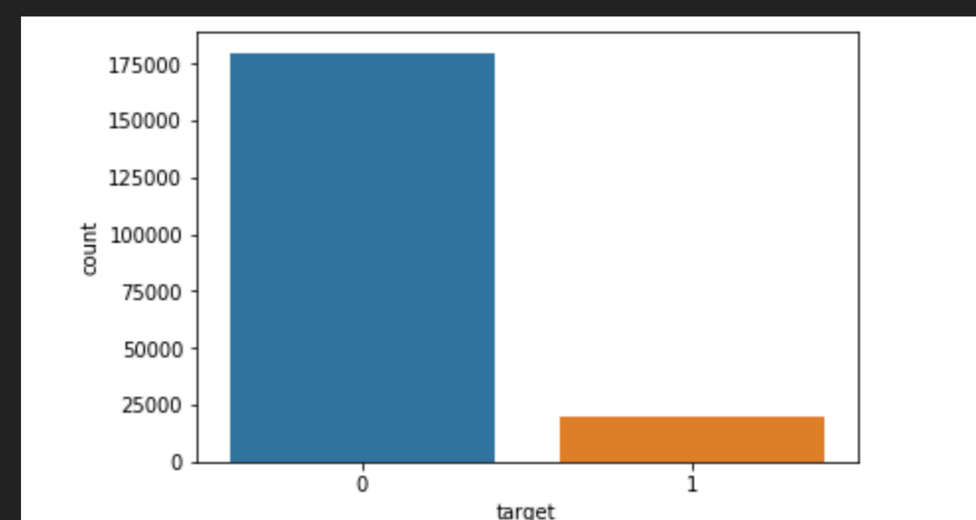
DATASET

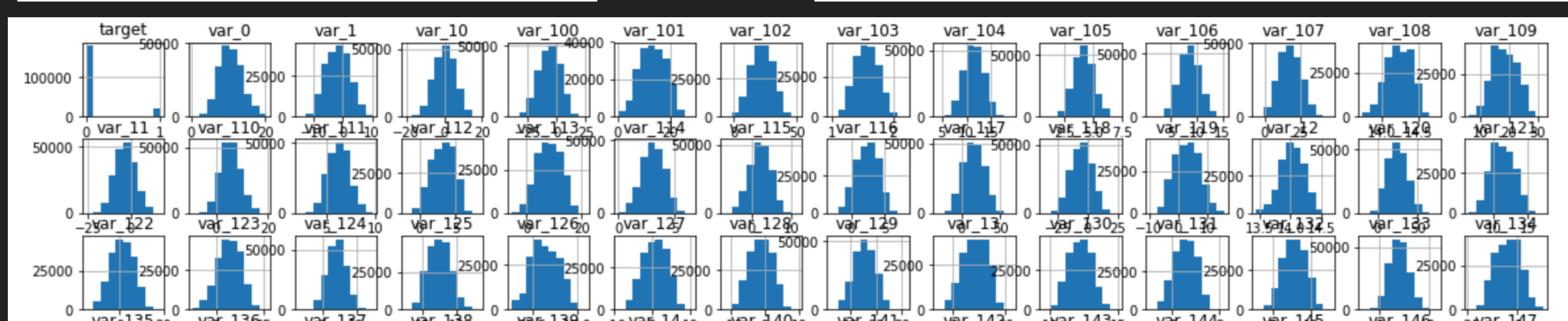
- Anonymized real world customer data from Santander bank
- Training data available in CSV format.
- Training data consists of 200K+ observations and 202 features.

EXPLORATORY DATA ANALYSIS

- Available data is pre-processed - Features are normally distributed
- Absence of correlation between the features.
- Imbalanced target class







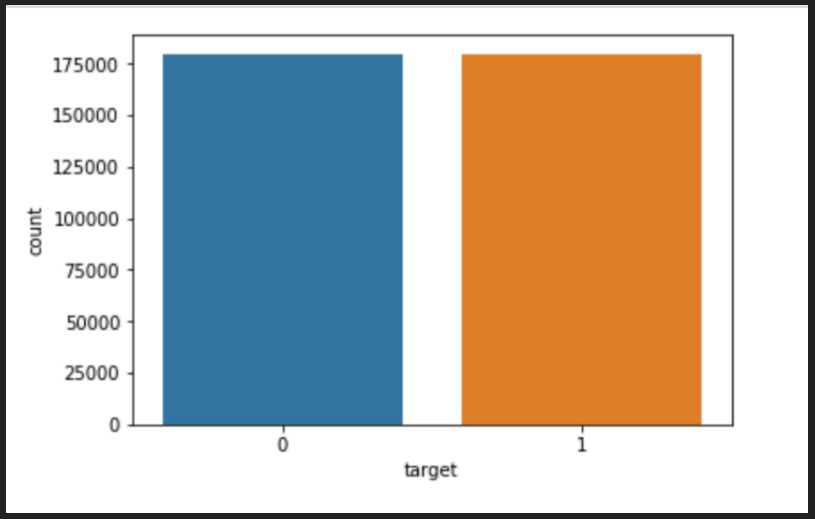
IMPLEMENTATION METHODOLOGY : DATA PRE-PROCESSING

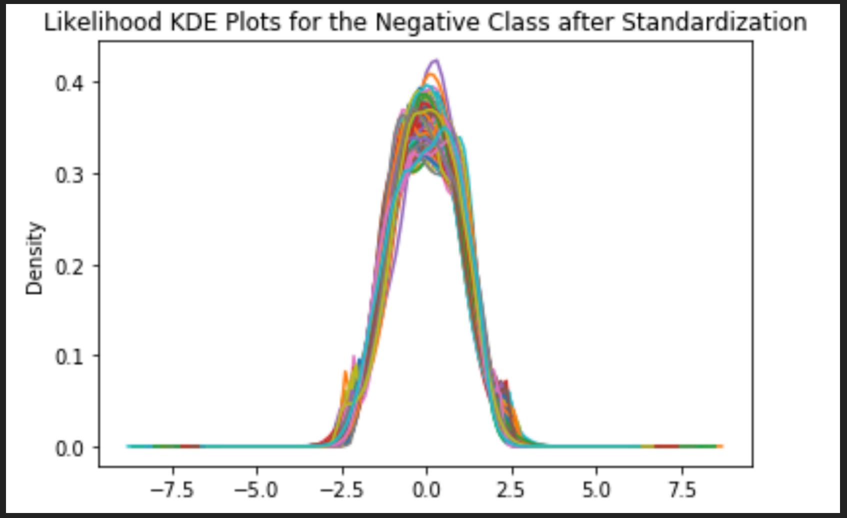
- Handling imbalanced data using SMOTE.
- Scaled features.

```
Before OverSampling, counts of label '1': 20098
Before OverSampling, counts of label '0': 179902

After OverSampling, the shape of train_X: (359804, 200)
After OverSampling, the shape of train_y: (359804,)

After OverSampling, counts of label '1': 179902
After OverSampling, counts of label '0': 179902
```

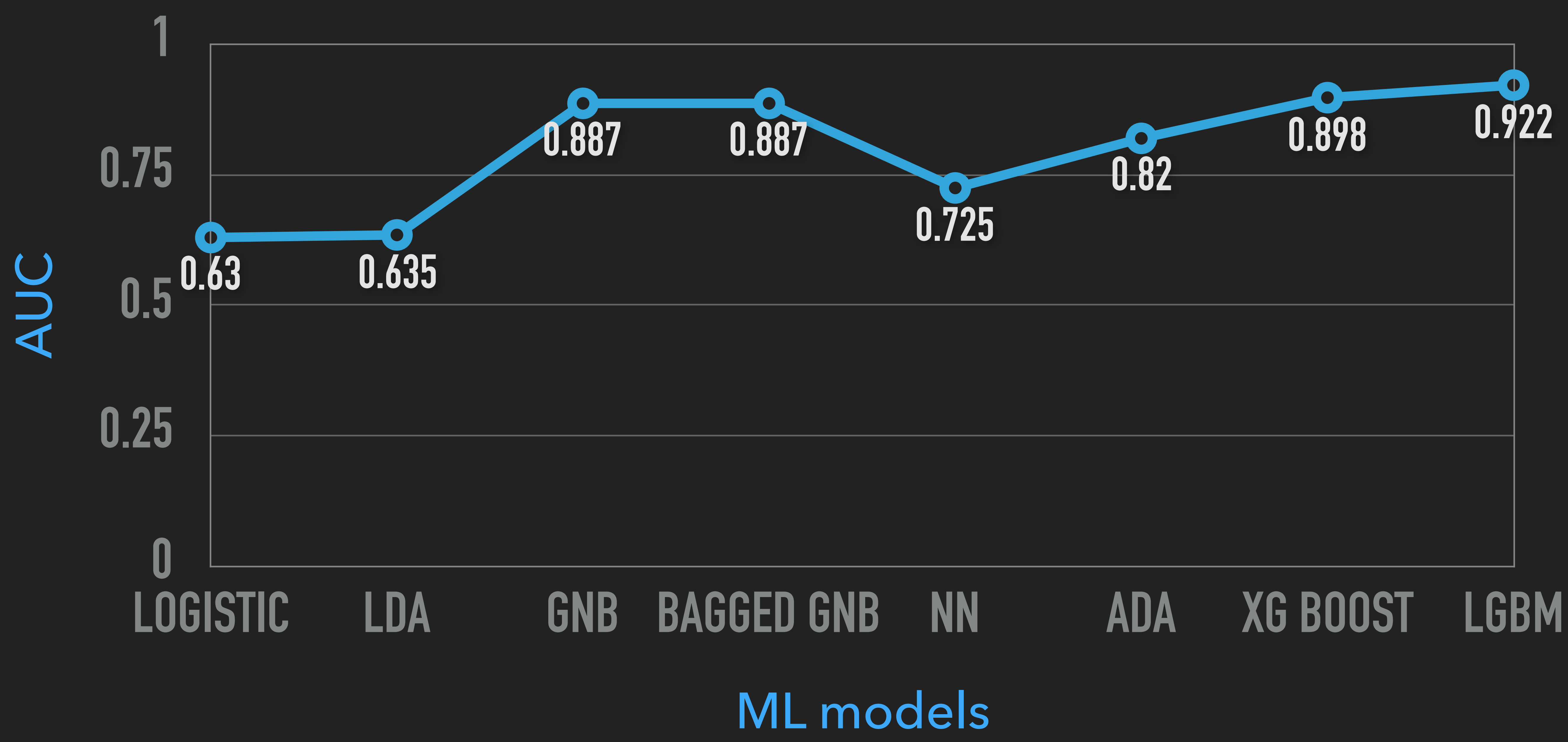




MODEL DEVELOPMENT

- **Base line models**
  - Logistic Regression - initial score of 0.630
  - EDA pointed towards Gaussian Naive Bayes - initial score of 0.887
- **Neural Networks**
  - Feedforward Neural Network - Best of 0.725
- **Ensemble methods**
  - Gaussian Naive Bayes - - Best of 0.888
- **Boosting Methods**
  - XGBoost, LGBM - Best of 0.922 AUC

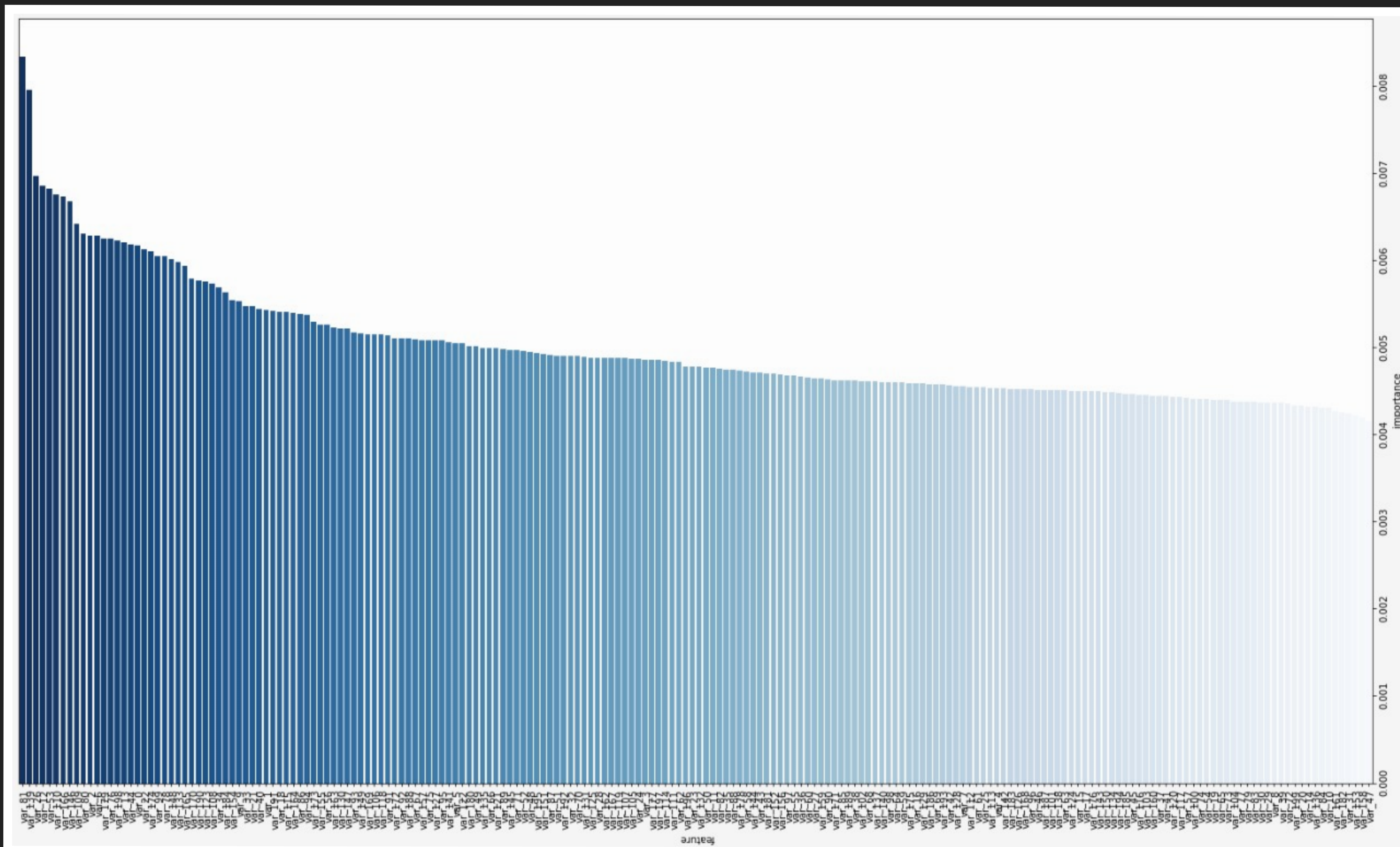
AUC REPORT


1
0.75
0.5
0.25
0
AUC
0.63
0.635
0.887
0.887
0.725
0.82
0.898
0.922
LOGISTIC
LDA
GNB
BAGGED GNB
NN
ADA
XG BOOST
LGBM
ML models

MODEL DEVELOPMENT

- Feed forward Neural Networks
  - ReLU activation
  - Learning rate = 1e-4
  - Iterations = 1000
  - Hidden layers = 3(x 13 neurons each)
- Boosting Methods
  - XGBoost
    - Learning rate = 0.01
    - Max depth = 2
    - Learning objective = binary logistic
  - LGBM
    - Boosting type = Gauss
    - Learning rate = 0.05
    - Max depth = 5
- Parameter tuning using Grid search

FEATURE IMPORTANCE



KAGGLE SUBMISSIONS AND SCORE EVALUATIONS

20 submissions for deepanshu_parihar

Sort by: Most recent

All | Successful | Selected

Submission and Description	Private Score	Public Score	Use for Final Score
NN3_balanced.csv 4 hours ago by deepanshu_parihar Neural Network on balanced data	0.72067	0.72514	☐
submission.csv 4 hours ago by deepanshu_parihar LightGBM model using gradient boosted decision tree	0.92089	0.92244	☐
submission_goss.csv 16 hours ago by deepanshu_parihar add submission details	0.92011	0.92195	☐
submission_dart.csv 17 hours ago by deepanshu_parihar add submission details	0.92028	0.92196	☐
submission.csv 18 hours ago by deepanshu_parihar add submission details	0.92089	0.92244	☐
LGBMt.csv 20 hours ago by deepanshu_parihar add submission details	0.60421	0.60453	☐

Overview | Data | Kernels | Discussion | Leaderboard | Rules | Team | My Submissions | Late Submission

Submission and Description	Private Score	Public Score	Use for Final Score
ensemble_xgboost_3.csv 2 days ago by Ajeya Kempegowda With upsampled data	0.79830	0.79677	☐
ensemble_xgboost_2.csv 2 days ago by Ajeya Kempegowda Tuned params	0.89681	0.89881	☐
ensemble_xgboost.csv 3 days ago by Ajeya Kempegowda XGB - Learning rate tweak	0.87246	0.87528	☐
ensemble_xgb.csv 3 days ago by Ajeya Kempegowda Base line XGB	0.36061	0.35522	☐
ensemble_ada.csv 3 days ago by Ajeya Kempegowda Ensemble Boosting Classifier	0.82617	0.82939	☐
adaboost.csv 3 days ago by Ajeya Kempegowda ADA boost baseline	0.57835	0.58189	☐
ensemble_gnb.csv 11 days ago by Ajeya Kempegowda Ensemble model using GNB - test	0.68386	0.68358	☐
ensemble_gnb.csv 11 days ago by Ajeya Kempegowda Ensemble model using GNB	0.88763	0.88848	☐
qda1.csv 25 days ago by Ajeya Kempegowda	0.51428	0.51498	☐

NEURAL ODE - INTRODUCTION

- Parameterizes the derivative of hidden state layers.
- Provides a continuous depth model.
- Properties we understood
  - Memory efficiency: Constant memory cost wrt depth
  - Adaptive computation: Adapt error levels for accuracy
  - Continuous time series model: For time series model (unlike RNN's)

NEURAL ODE – IMPLEMENTATION

- ‘odeint’ interface to solve the initial value problem(ODE+initial state)
- odeint(ODE solver) tries to find the trajectory satisfying the ODE that passes through the initial conditions.
- ODE solver can be tweaked to acquire fixed steps(Euler) or to adaptive(Runge Kutta)
- Back-propagation is done using ‘ode_adjoint’ that solves adjoint ODE in O(1) space complexity.