B555 Machine Learning Class Project

Aniket Gaikwad and **Dileep Viswanathan**

Introduction

In this report we discuss the analysis of various machine learning algorithms on the “Don’t get kicked” dataset in kaggle.com. This dataset contains information about used vehicles sold at auctions. The task is to predict if the vehicle to be sold at an auction will be a good buy or not. A vehicle that is not in the best of conditions is considered a kick. There are about 34 features provided of which most are categorical like the size, model, make, etc. Discussed below are the details on the dataset and the various approaches used by us at classification.

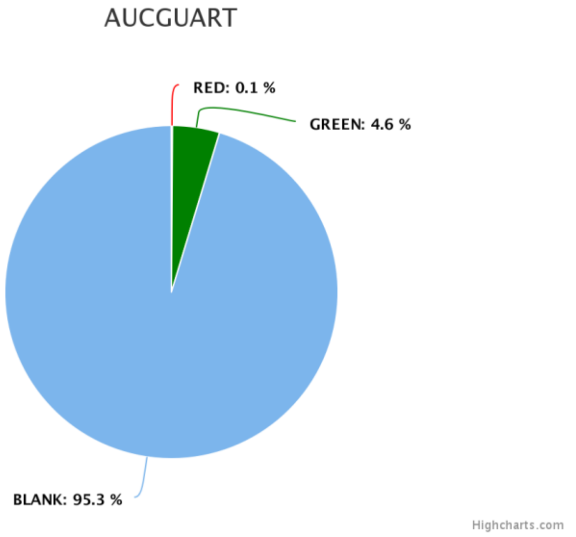
Data

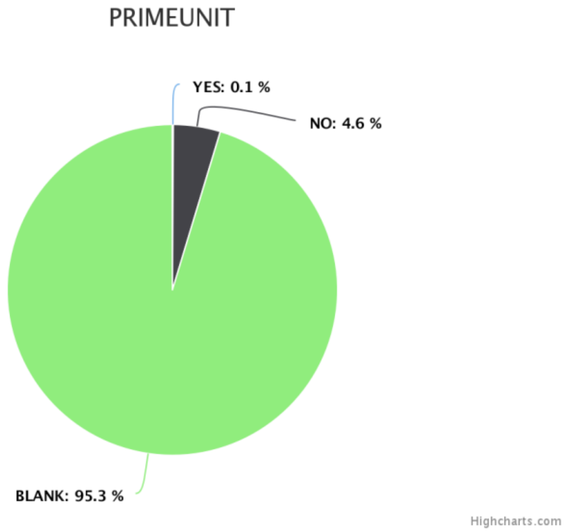
The dataset has 34 features with just over 72,900 data points. The binary target label *IsBadBuy* can take values 1 and 0 with 1 indicating the vehicle to be a kick. Every data point is uniquely identified by the *RefId* column. The features are categorical, numeric and date type. Information on the age and usage of the vehicle is given by *PurchDate*, *VehicleAge*, *VehYear* and *VehOdo.* Brand details are described by *Model*, *Make*, *Trim* and *SubModel*. Guarantee id specified by *AUCGUART* and demand by *PRIMEUNIT*. Geographical information is provided by *VNZIP* and *VNST* which are the zip code and state in which the vehicle was purchased respectively. Eight Manheim Market Report (MMR) prices describe the best estimate of the market price for the vehicle. The buyer id of the vehicle is given by *BYRNO* while the other features *IsOnlineSale*, *WarrantyCost* and *VehCost* speak for themselves.

Analysis

Features removed

- *SubModel* and *Model* have a very large range of categories
- *PRIMEUNIT* and *AUGGUART* have a large number of missing values

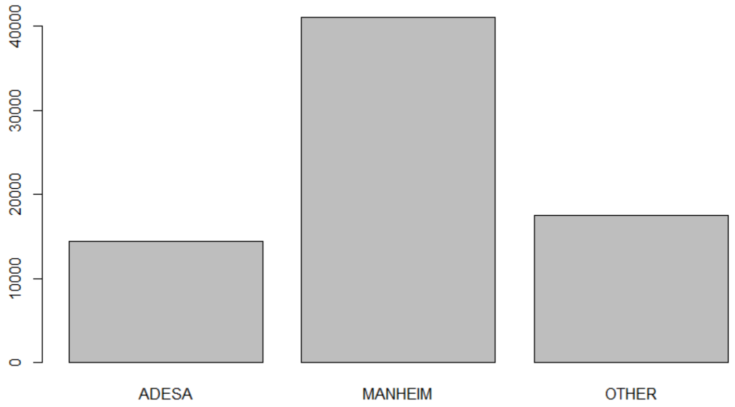




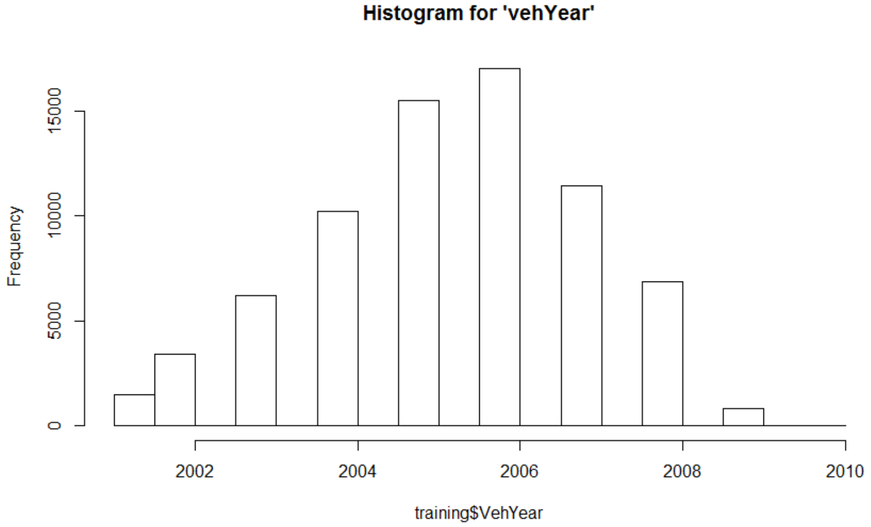
- Preliminary results on date of purchase, *PURCHDATE*, did not support our belief that it would be a good feature for prediction
- Zip code information *VNZIP1* is too specific to an area and becomes a sparse attribute
- *BYRNO,* which indicates the buyer of the vehicle, does not add any value to the feature set

Features selected

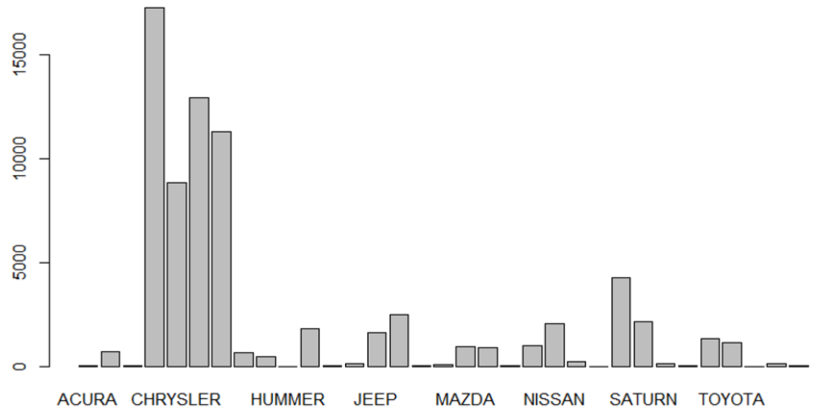
- *Auction* has only 3 discrete values which are decently well spread



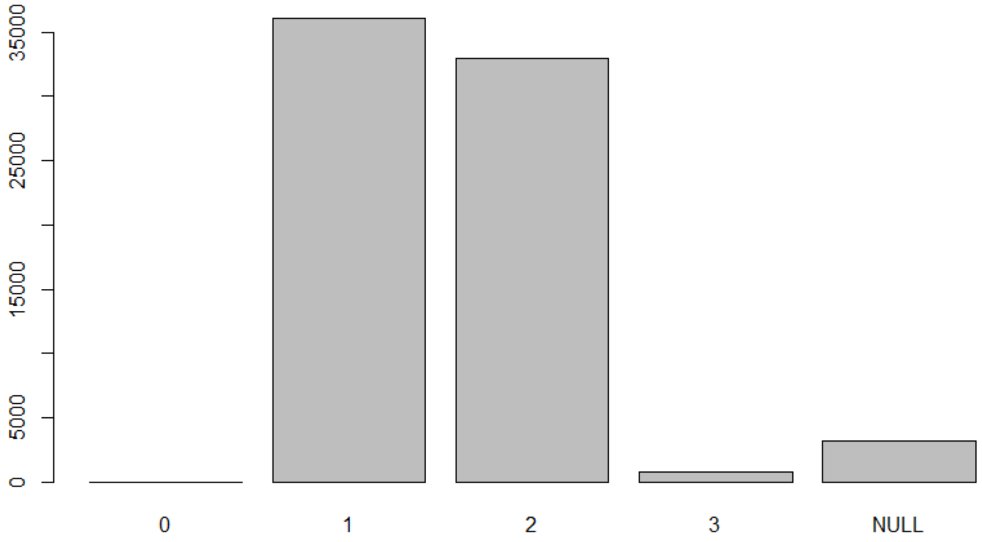
- *VehYear*, indicating the year of make of the vehicle, follows a Gaussian distribution



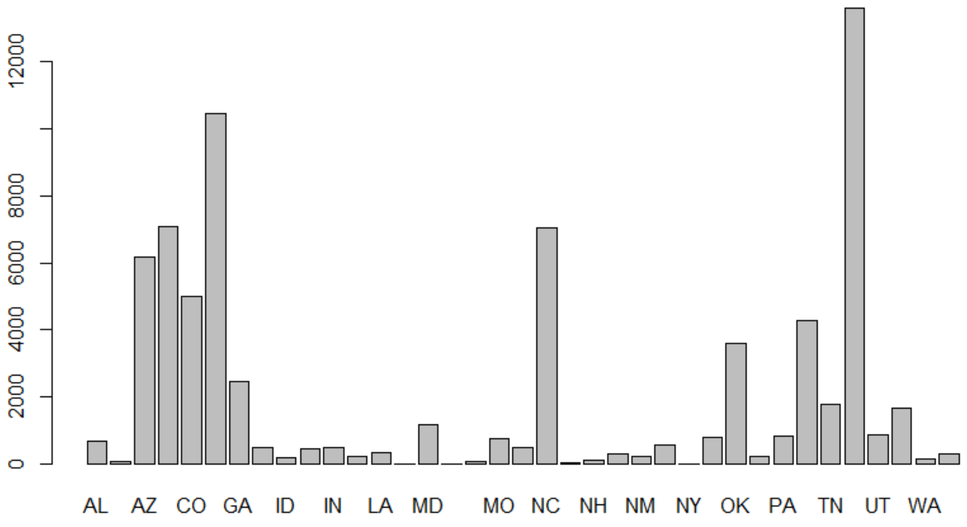
- *Make* is left skewed and we believe that the manufacturer would have an impact on the resale price



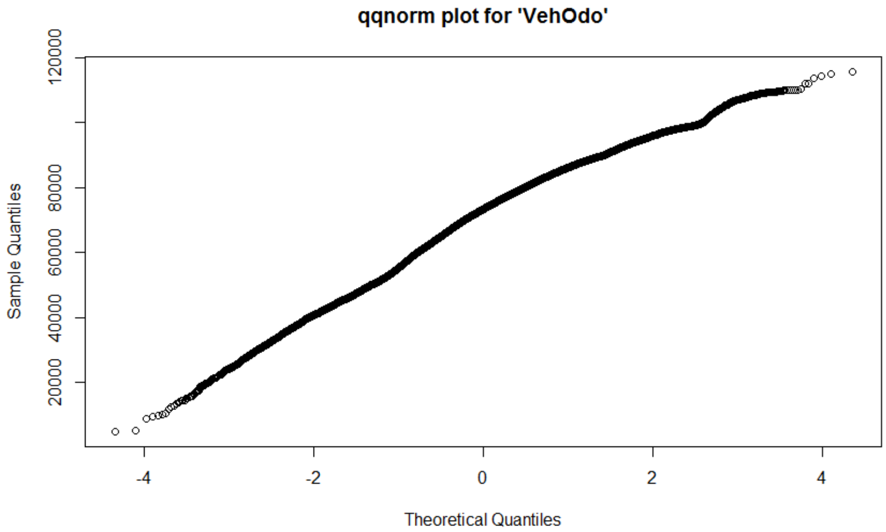
- *Color* might be useful because certain colors are preferred by a large group of people
- *WheelTypeID* has a similar distribution between two values



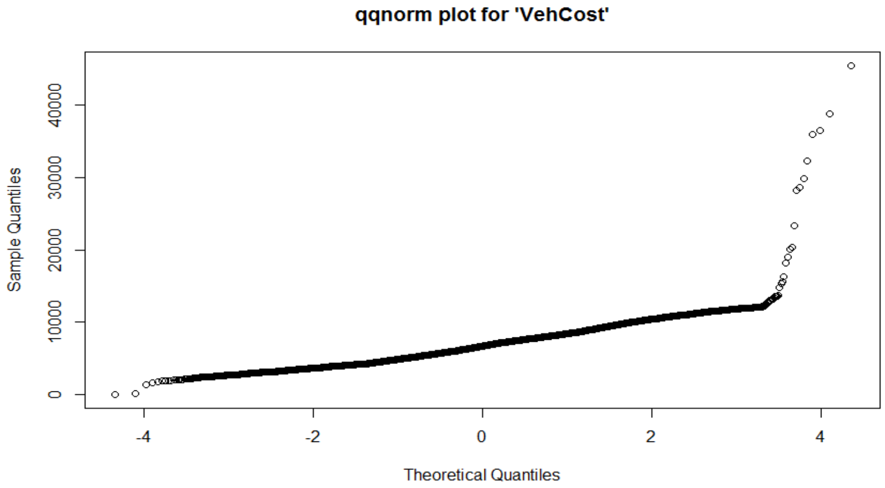
- *Nationality* might indicate that people from certain countries could be poor maintainers of vehicles
- *VNST* indicating the states is neither skewed nor normally distributed



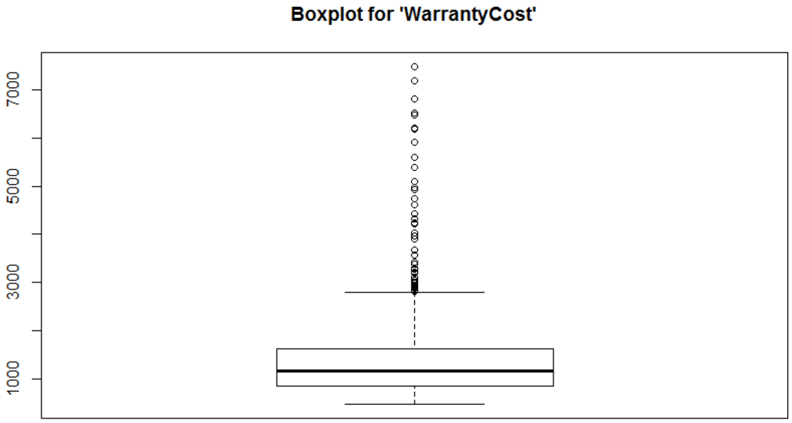
- *MMR prices* are continuous values which directly speak about the estimated market value
- *IsOnlineSale* could have an impact considering a common perception that online purchases are more convenient for users
- *VehOdo* follows a Gaussian distribution



- *VehAge* follows a normal distribution
- VehCost being right skewed with outliers can be carefully sampled to get normally distributed values

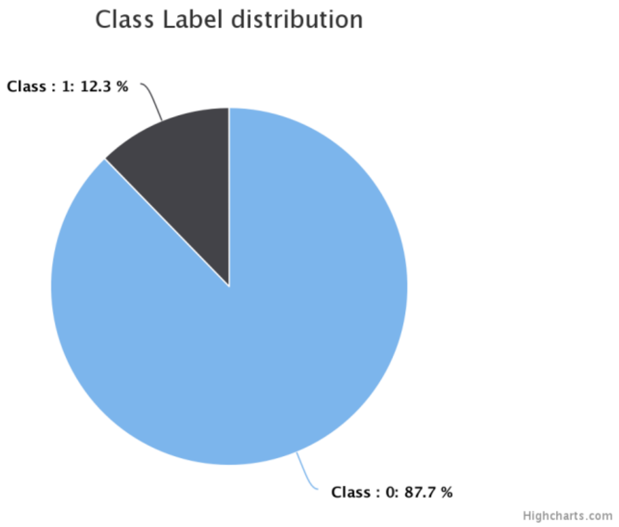


- *WarrantyCost* is right skewed



Class Imbalance

Out of the 72,985 data points, 64008 are negatives and 8977 are positives indicating than 87.7% of the data points are negatives.



In order to handle the imbalance, we sampled positives and negatives in the ratio 2:1. Using the entire dataset without sampling would result in the majority class being predicted for every data point, resulting in a higher precision but that is a false sense of achievement.

Preprocessing

- Categorical features such as *Auction*, *Model*, *Trim*, *Color*, *Transmission*, *Nationality*, *Size*, *TopThreeAmericanName* and *VNST* have non-integer values. So we mapped them to integer values and modified the dataset
- Replaced null values with 100
- As the integer replacements of strings and null values would have an unnecessary impact on data when we compute the Euclidean distance, the attributes are vectorized. When an attribute is vectorized, it is split into multiple attributes, each representing a distinct value of the vectorized feature
- sklearn.preprocessing.OneHotEncoder has been used for vectorization

Classifiers

We started with 4 classifiers namely Gradient Boost, Logistic Regression, Linear SVM and Gaussian SVM. For each classifier we select 5 hyper-parameters leading to 20 models in total. We trained and tested each model on a sample to decide on two classifiers that might give the best results on the vehicle dataset.

- Gradient Boost
    - Hyper-parameter (number of boosting stages) values – 2, 3, 5, 10, 100
    - For 2 and 3 all predictions were negative and for 5, 10 and 100 all predictions positive
    - This was due to class imbalance
- L2 Logistic Regression
    - Hyper-parameter (regularization factor) values – 0.001, 0.01, 0.1, 1, 10
    - We obtained similar results for all hyper-parameter values
- Linear SVM
    - Hyper-parameter (number of centers) values – 20, 50, 100, 200, 300

    - Results for all parameter values were similar but not promising
- Gaussian SVM
    - Hyper-parameter (number of centers) values – 20, 50, 100, 200, 300
    - Results looked reasonably better than the other classifiers

Based on the above observations and class label distributions, we decided to eliminate Gradient Boost as it predicted either positive or negative for all test samples and this can be attributed to the class imbalance. As many features follow a normal distribution we preferred Gaussian SVM over Linear SVM. Logistic Regression performed better than Linear SVM on the sample data to select the best classifiers.

Experiments

Part 1

- Weighted learning – a weight of 10 was assigned to the positive (minority) class and 1 to the negative class
- Learned models using Gaussian SVM and Logistic Regression with 5 hyper-parameters
- 10-fold cross validation was performed

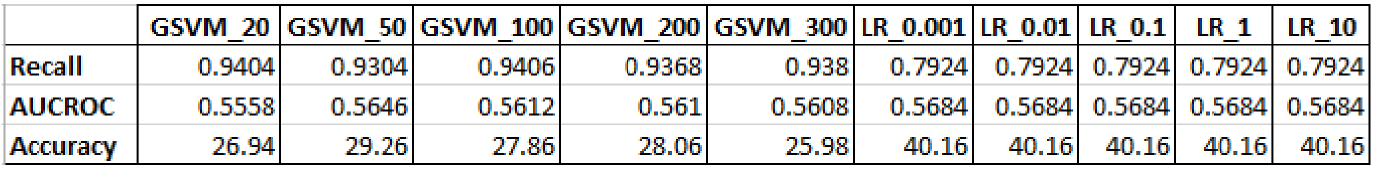
Part 2

- Data sampling – we sampled positives and negatives in the ratio 1:2 to tackle class imbalance
- Learned Gaussian SVM with 100 centers and L2 Logistic Regression with regularization factor 0.1
- The above parameters were selected based on results from Part 1
- Performed 10-fold cross validation

Due to class imbalance we chose AUC-ROC, Recall and F1 scores as the evaluation methods over accuracy.

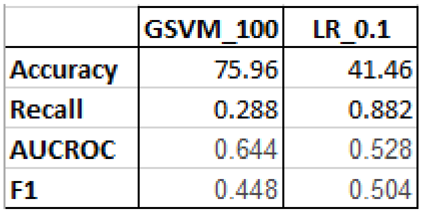
Results

Part 1

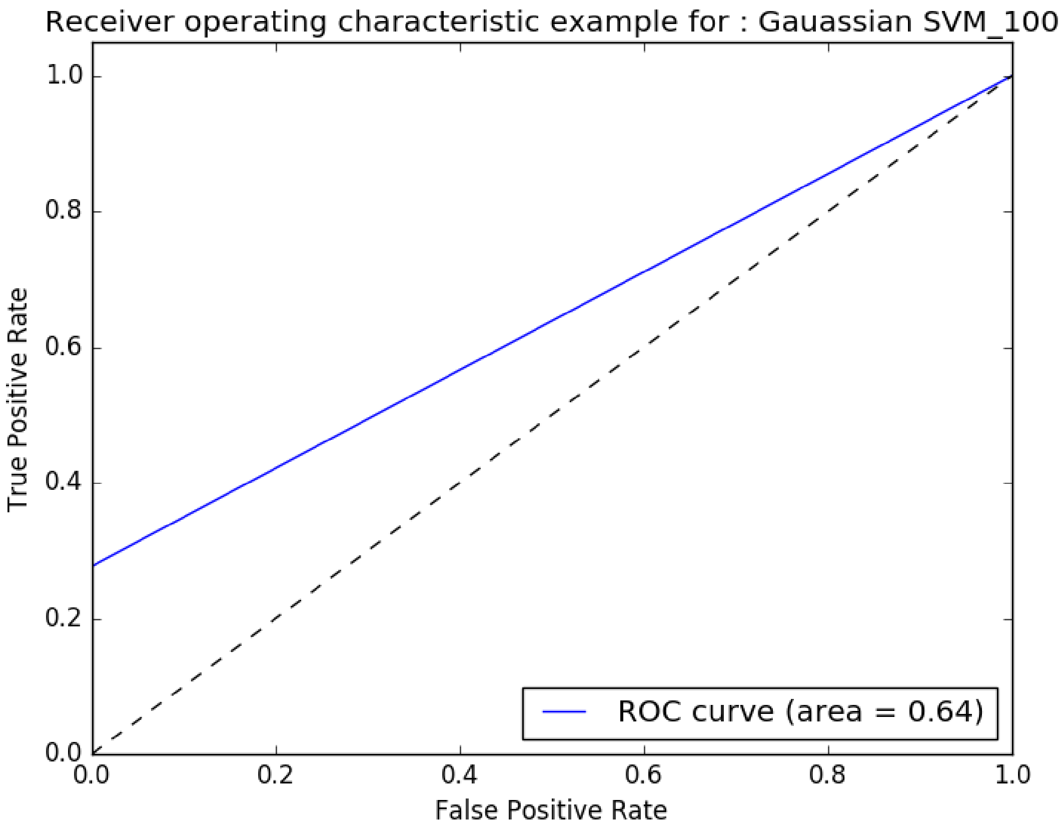


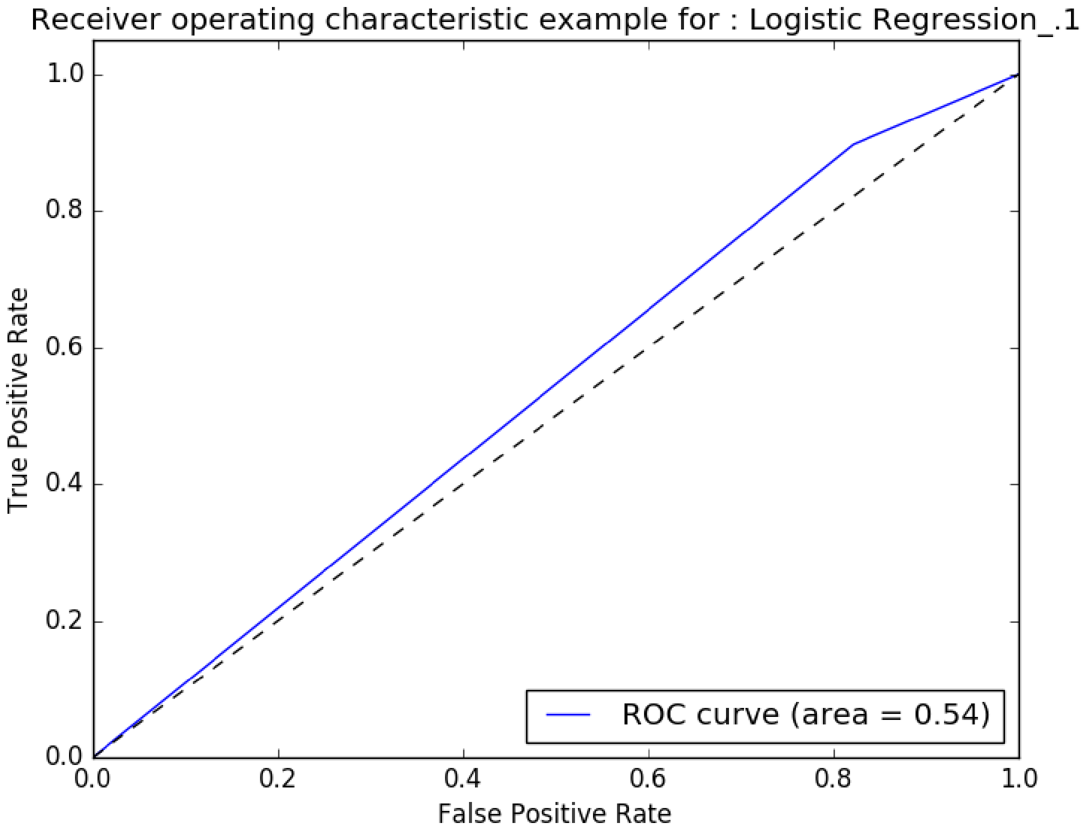
	GSVM_20	GSVM_50	GSVM_100	GSVM_200	GSVM_300	LR_0.001	LR_0.01	LR_0.1	LR_1	LR_10
Recall	0.9404	0.9304	0.9406	0.9368	0.938	0.7924	0.7924	0.7924	0.7924	0.7924
AUCROC	0.5558	0.5646	0.5612	0.561	0.5608	0.5684	0.5684	0.5684	0.5684	0.5684
Accuracy	26.94	29.26	27.86	28.06	25.98	40.16	40.16	40.16	40.16	40.16

Part 2



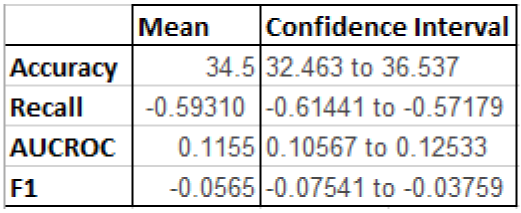
	GSVM_100	LR_0.1
Accuracy	75.96	41.46
Recall	0.288	0.882
AUCROC	0.644	0.528
F1	0.448	0.504


Receiver operating characteristic example for : Gauassian SVM_100
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.64)


Receiver operating characteristic example for : Logistic Regression_.1
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.54)

Hypothesis Testing

We performed t-test on all evaluation measures for the results we got on Gaussian SVM and Logistic Regression after we performed 10-fold cross validation. For all the 4 tests the p-value was less than 0.0001, so with respect to 95% confidence interval, we reject the null hypothesis that the two classifiers are not statistically significantly different in performance. Based on the mean value we observe that Gaussian SVM is statistically significantly better than Logistic Regression.



	Mean	Confidence Interval
Accuracy	34.5	32.463 to 36.537
Recall	-0.59310	-0.61441 to -0.57179
AUCROC	0.1155	0.10567 to 0.12533
F1	-0.0565	-0.07541 to -0.03759

Conclusion

In Part 1 we focussed on 'recall' as data was class imbalanced (87% - Negative and 13% - Positive). So, it was essential to build a model that is able to predict postive example correctly. So, 'recall' which is True_Positive/(True_Positive+False_Positive), gives us the measure to evaluate results we got.

In Part 2, we had class balanced data (66% - Negative and 34% - Positive).So,our analysis was focussed on 'accuracy'.

As can be observed, for Part 1, Gaussian SVM gives a higher recall as compared to Logistic Regression. This result can be bolster by fact that many of the features used are drawn from 'Guassian Distribution' and theoriotically combination of Guassian distribution results in Guassian distribution. Also, Logistic regression classify based on log-likelihood & log-likelihood favors the majority class. This implies that, for Logistic regression, it will be most predictions as negative(majority class) giving better accuracy but less recall. So, results we got are in harmony with theory.

In Part 2 of our experiment, our aim was to build a model that will handle data which will be balanced (As Kaggle test data is hidden we had to assume that our model may/may not be tested on balanced data). So,as per experiment we performed, because of balanced data, we more focussed on 'accuracy'. The results we got shows that in this case also, Guassian SVM is better model.

References

[1] Python sklearn packages

[2]https://www.researchgate.net/publication/262523736_Data_Science_with_Kaggles_Competition_Dont_Get_kicked%21

[3] Kaggle.com

[4] git@github.iu.edu:diviswan/B555-Project.git