NIFTY 50 Direction - Supervised Learning - Classification

Kushagra Singhal

2021-09-18

1 Objective

The objective of this report is to present the classification of Nifty 50 (Indian Stock Market Index) next day direction given Open Interest (OI) data. We will also touch upon interpretation of the model in terms of different predictors used.

2 Dataset and Attributes

The dataset contains information about OI and change in OI at different strikes for 627 days. The set of features included are OI for different strikes at and around the at-the-money (ATM) strike, and also the changes in these OI. The aim of this project is to learn a model which classifies the next day Nifty50 return into positive or negative classes. Positive and negative classes are represented by 1 and 0 respectively.

The information about data columns is also provided in table 1. As can be seen, there are total 40 features and 1 target column called y. The data types are a mix of integer, and floating. There are a total of 627 samples in the dataset.

To explain the features further, $CE100p\_oi$ means OI for a call option 100 points above ATM strike, whereas $CE100n\_oi$ would imply 100 points below. The features with d in the name imply the daily change. Figure 1 shows the distribution of up vs down moves. Figure 2 shows the histogram of correlations between different pair of features.

Table 1: Information about columns in the dataset

	Column	Non-Null Count	Dtype
0	CE400m_oi	627	float64
1	PE400m_oi	627	float64
2	CE300m_oi	627	float64
3	PE300m_oi	627	float64
4	CE200m_oi	627	float64
5	PE200m_oi	627	float64
6	CE100m_oi	627	float64
7	PE100m_oi	627	float64
8	CE0m_oi	627	float64
9	PE0m_oi	627	float64
10	CE0p_oi	627	float64
11	PE0p_oi	627	float64
12	CE100p_oi	627	float64
13	PE100p_oi	627	float64
14	CE200p_oi	627	float64
15	PE200p_oi	627	float64
16	CE300p_oi	627	float64
17	PE300p_oi	627	float64
18	CE400p_oi	627	float64
19	PE400p_oi	627	float64
20	CE400m_d_oi	627	float64
21	PE400m_d_oi	627	float64
22	CE300m_d_oi	627	float64
23	PE300m_d_oi	627	float64
24	CE200m_d_oi	627	float64
25	PE200m_d_oi	627	float64
26	CE100m_d_oi	627	float64
27	PE100m_d_oi	627	float64
28	CE0m_d_oi	627	float64
29	PE0m_d_oi	627	float64
30	CE0p_d_oi	627	float64
31	PE0p_d_oi	627	float64
32	CE100p_d_oi	627	float64
33	PE100p_d_oi	627	float64
34	CE200p_d_oi	627	float64
35	PE200p_d_oi	627	float64
36	CE300p_d_oi	627	float64
37	PE300p_d_oi	2 627	float64
38	CE400p_d_oi	627	float64
39	PE400p_d_oi	627	float64
40	y	627	int64

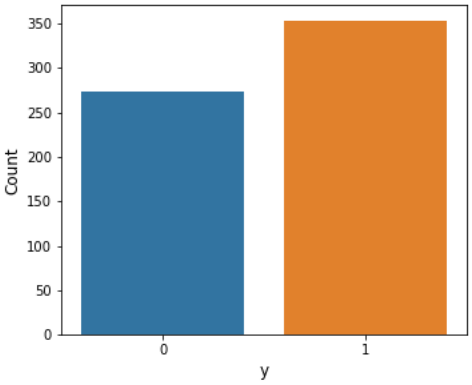

Figure 1: Barplots for up(1) and down(0) move

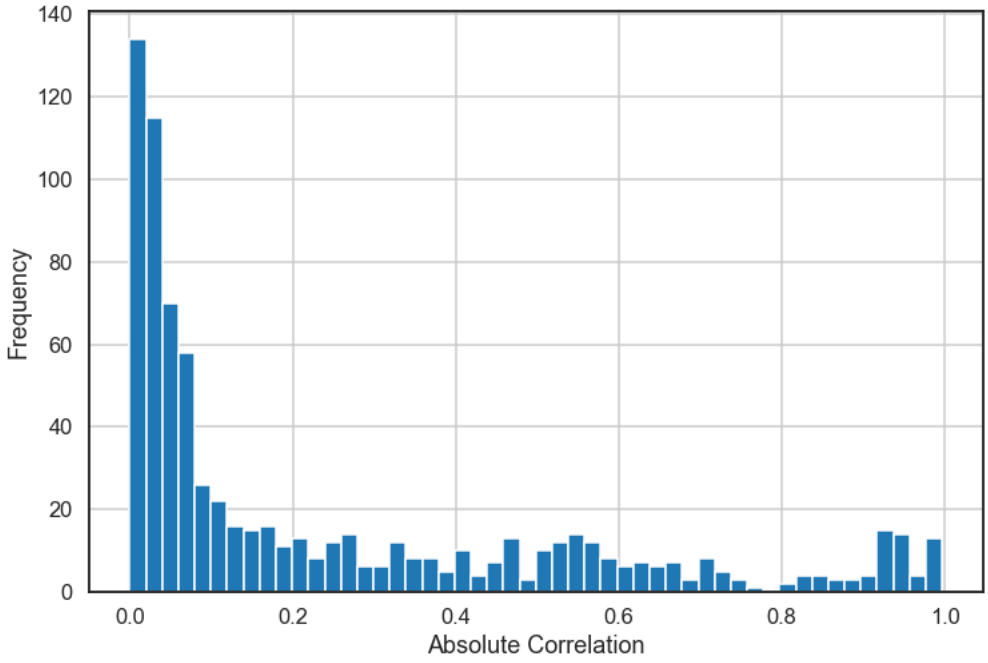

Figure 2: Correlation histogram between features

2.1 Data Cleaning and Feature Engineering

As the dataset is created by me using the data available at NSE website, there are no missing values for any of the features or the target. As such, cleaning is no needed. However, we did perform some feature engineering. Rather than using the gross values of OI and OI change, we convert them into fractions. For example, for features representing the OI, the final feature will be the fraction of total OI on that particular day. Similarly, the OI change feature finally denotes the fraction of change of the total OI change that day. After this scaling, no other feature engineering is performed.

Table 2: Summary of results for Logistic Regression Models

	lr	l1	l2
precision	0.504233	0.504233	0.504233
recall	0.539683	0.539683	0.539683
fscore	0.477764	0.477764	0.477764
accuracy	0.539683	0.539683	0.539683
auc	0.498125	0.498125	0.498125

3 Classification Models

In this section, we will look at three classification models. First, is the logistic regression model which serves as a baseline model. Second, we look at a KNN classifier model, followed by SVM and decision tree models.

3.1 Baseline - Logistic Regression

The most straightforward model is a logistic regression models without any penalty, l1 penalty and l2 penalty. This was implemented using sklearn in python. The table 2 shows the precision, recall, f-score, accuracy and AUC of the different logistic regression models. As is observed, the models do not perform any better than random guesses. Figure 3 shows the confusions matrix for the models. Figure 4 shows the ROC and PC curves.

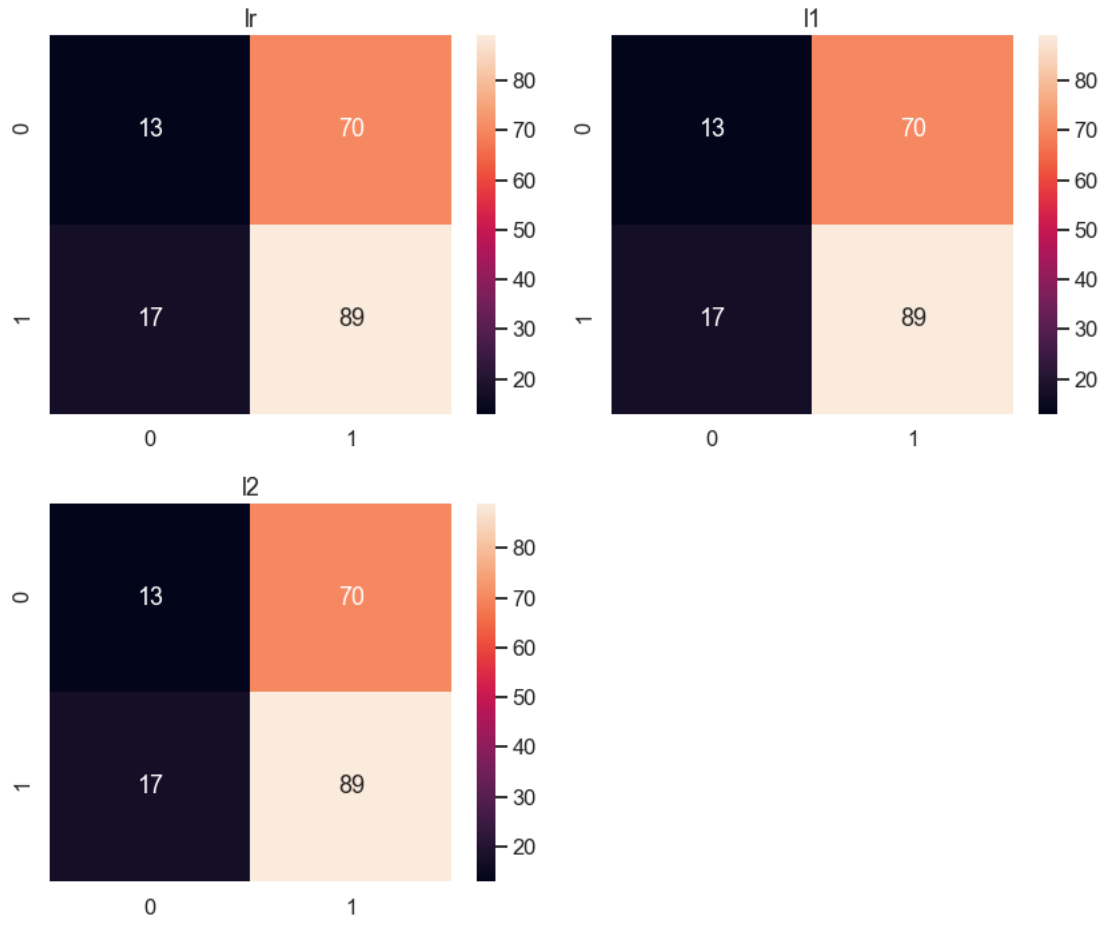

Figure 3: Confusion Matrices for logistic regressions

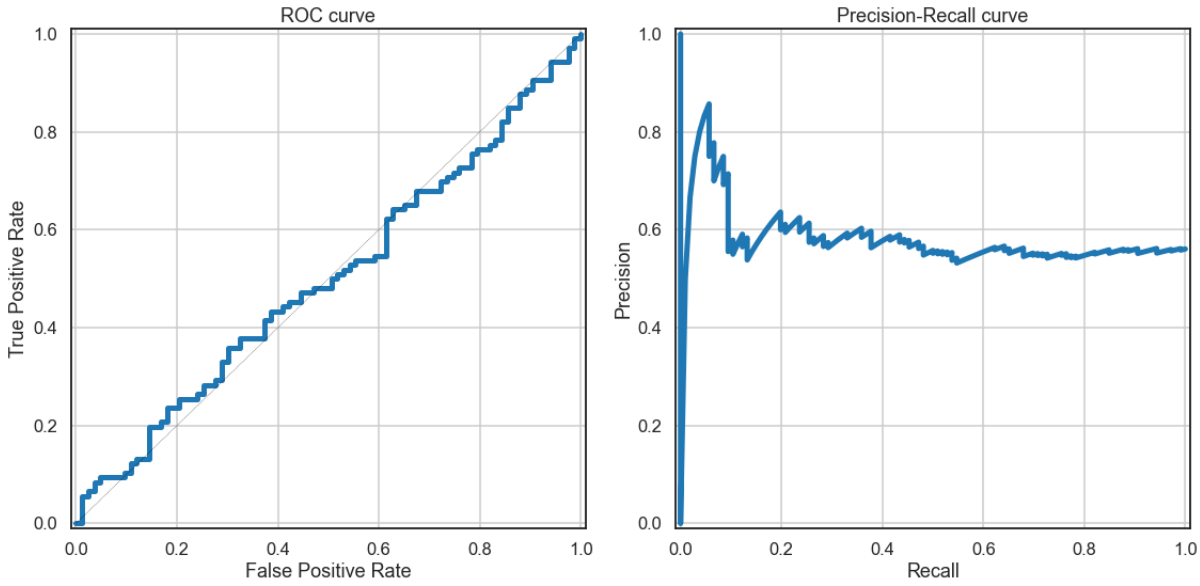

Figure 4: ROC and Precision-Recall Curves for logistic regressions

3.2 KNN Classifier

The next model explored is the KNN classifier. We tried to find the optimal number of neighbors using the elbow method as shown in figure 5. We selected k=29 based on this analysis. Figure 6 shows the confusion matrix, Figure 7 shows the ROC and PC curves. Table 3 shows the summary stats for the model. It is observed that KNN classifier performs much better than the baseline model.

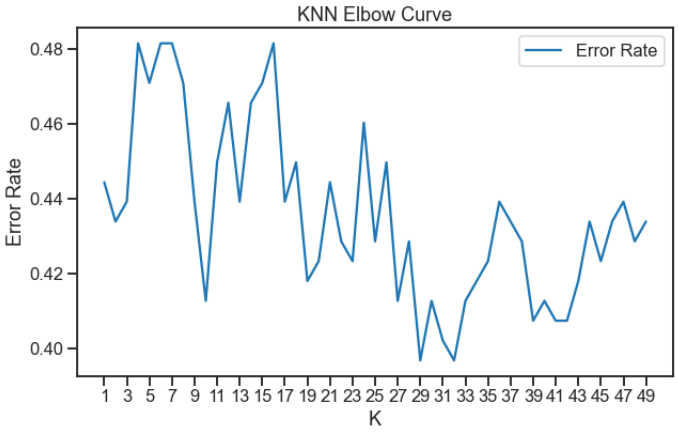

Figure 5: KNN elbow method analysis

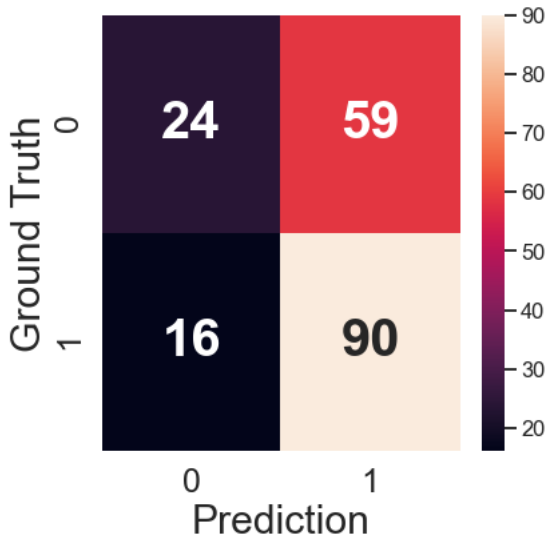

Figure 6: Confusion Matrices for KNN classifier

Table 3: Summary of results for KNN Classifier Model

	Class 0	Class 1
precision	0.62	0.61
recall	0.29	0.86
fscore	0.39	0.71
overall accuracy	0.61	
overall auc	0.57	

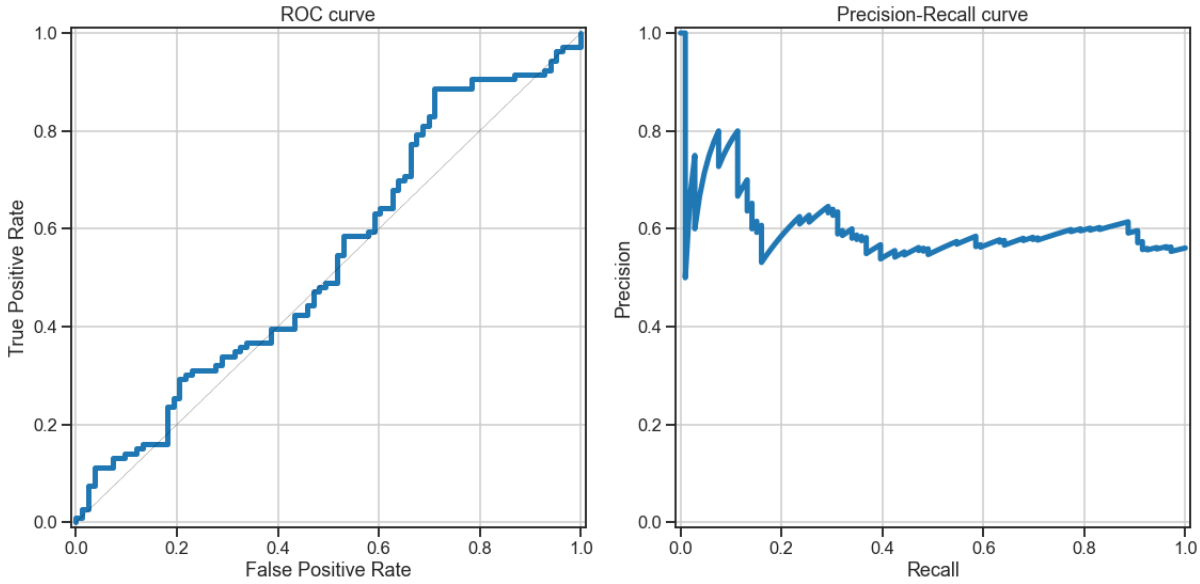

Figure 7: ROC and Precision-Recall Curves for KNN classifier

3.3 SVM

We then move to the SVM model with radial basis function and gamma = 10. Figure 8 shows the confusion matrix, Figure 9 shows the ROC and PC curves. Table 4 shows the summary stats for the model. It is observed that SVN performs much better than the baseline model but worse than KNN.

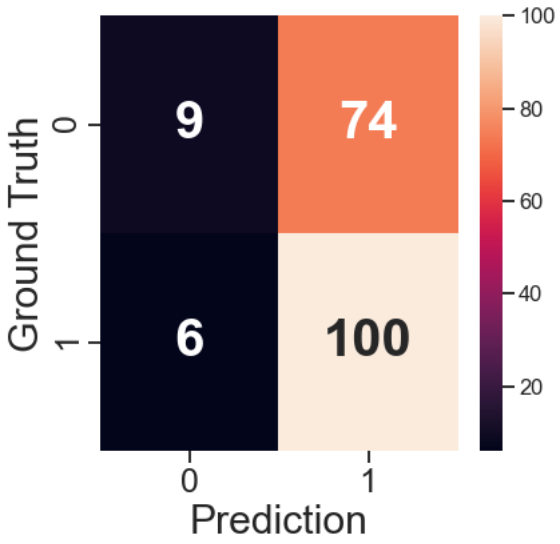

Figure 8: Confusion Matrices for SVM classifier

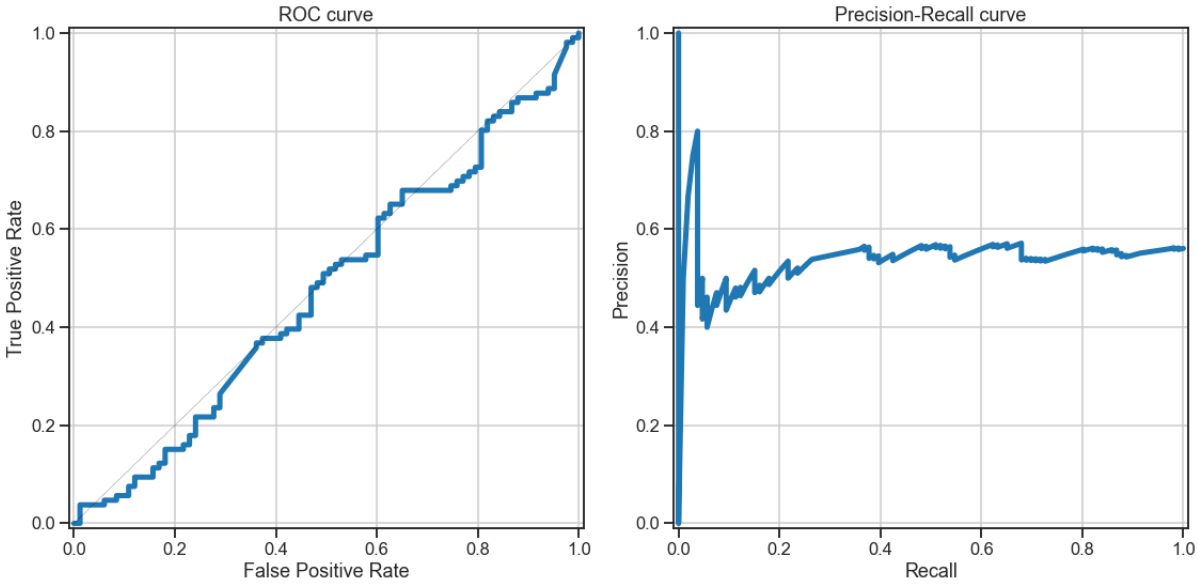

Figure 9: ROC and Precision-Recall Curves for SVM classifier

Table 4: Summary of results for SVM Classifier Model

	Class 0	Class 1
precision	0.6	0.57
recall	0.11	0.94
fscore	0.18	0.71
overall accuracy	0.58	
overall auc	0.52	

3.4 Decision Tree Classifier

Next we explore the decision tree model for classification. We run the GridSearch CV to find the best model first. The best estimator uses tree of depth 7. Figure 10 shows the confusion matrix, Figure 11 shows the ROC and PC curves. Table 5 shows the summary stats for the model. It is observed that decision tree performs much better than the baseline model but worse than KNN and SVM.

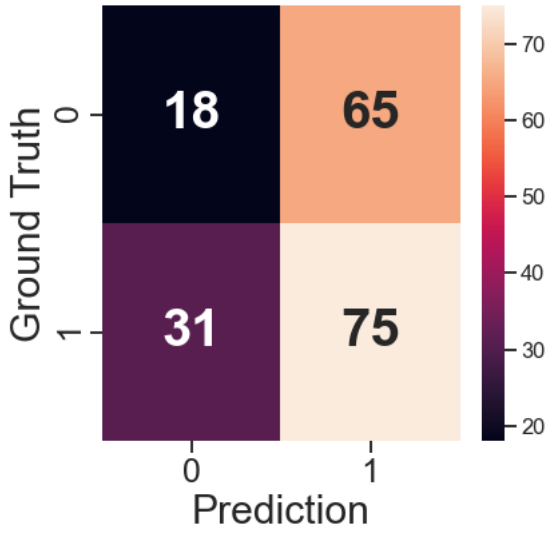

Figure 10: Confusion Matrices for decision tree classifier

Table 5: Summary of results for decision tree classifier Model

	Value
precision	0.54
recall	0.71
fscore	0.61
overall accuracy	0.49
overall auc	0.46

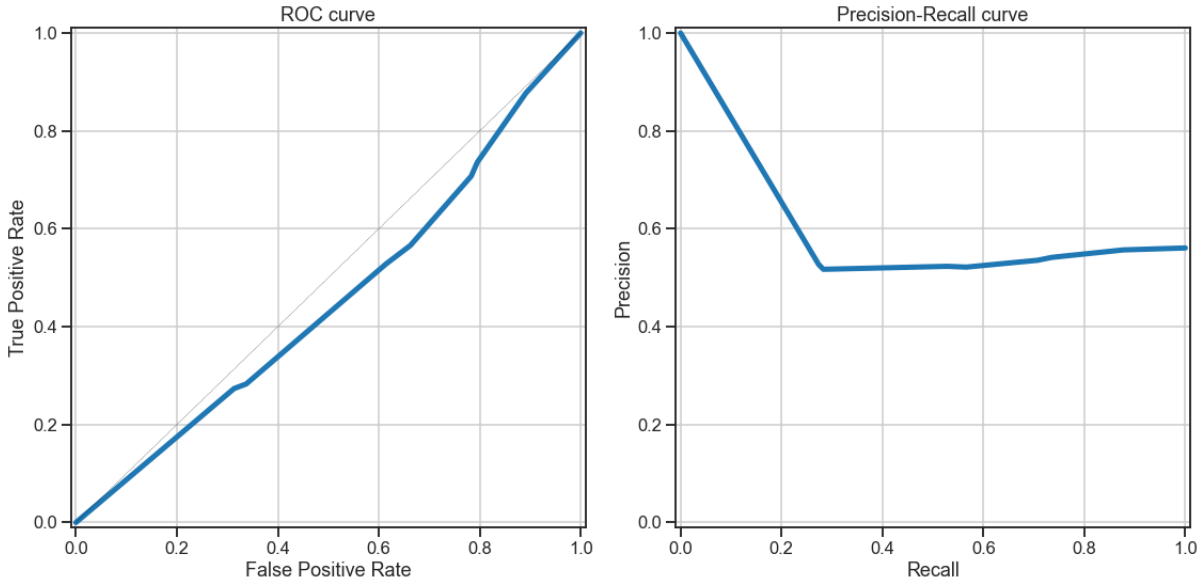

Figure 11: ROC and Precision-Recall Curves for decision tree classifier

4 Summary of Results

This section presents a summary of results obtained after training and testing the different models. The train test split is 70:30 and the splits are same across models.

The baseline model which is the logistic regression performs as good as random guessing. The SVM classifier is a bit better. The decision tree is not performing well. However, the KNN classifier performs the best.

5 Model Choice

We would choose the KNN classifier model as the final one. It performs the best in terms of the different metrics. Further, the KNN classifier is a simple model to implement and understand.

6 Key Findings and Insights

Overall, it is observed that the KNN classifier with 29 neighbors does a good job in terms of precision metric. For class 1, the recall is also good. Overall accuracy is much better than the random guess model. The prediction of stock market direction is a pretty hard problem because of so many unknowns and uncertainties. When i started this project I was sure I am not going to find a model which does a very good job. If it did, I would become rich, won't I.

7 Next Steps

Some of the next steps could be:

- Include more features like recent price returns
- Use more advanced models like random forests, bagging, boosting, ensemble
- Use other transformations on data like z-score