Business Conclusions

Summary

The first and foremost, the incoming dataset initially was biased against people with certain backgrounds regarding **the search rate**. After running the simulations, we saw that for the half of simulations, the true class was received as null. Since the reason was not specified by the client, we suppose that simulations with not null true class were received just for searched vehicles and null true class corresponds with non searched vehicles, otherwise the issue can be on the client's side with sending true values. The vehicles which true class is null, cannot be used to evaluate our model performance, but these observations can be useful for estimation how fair our model is, since we don't need true class for that. Most of the available searches were relative to Black (32%) and White people (69%), as for sex, mostly Men were searched (79%) and for ethnicity, most of drivers were identified as 'non applicable' (74%) and as Hispanic ethnicity (26%).

The total amount of received observations is 9900 counting 5000 observations with no results available, thus the evaluation was possible just for 4900 observations. The prediction of our model was correctly estimated for 3349 observations, therefore the accuracy of our prediction is 0.68. However, our model found contraband in 579 observations from 1761 where contraband actually existed, which means recall for true positive equals 0.33 unlike the foreseen 0.45, i.e., we were not able to meet the requirement 'A minimum 50% success rate for searches' (**at least with the given data sample**). The received precision of 0.61 has been met closer to the expected 0.69, i.e., 375 vehicles from 3145 were determined as having contraband, but actually they were innocent. Measuring the false positive rate is crucial for understanding how fair our model is. Since the incoming simulations had an uneven class distribution of true positives and negatives (we had more vehicles without contraband), let's check out how balanced are our precision and recall (whether our false predicted presence of contraband and false absence of it have similar cost) reviewing f1 score. We received 0.43 which is far apart from expected 0.67 and it means that predicted false absence of contraband is much higher than false predicted presence.

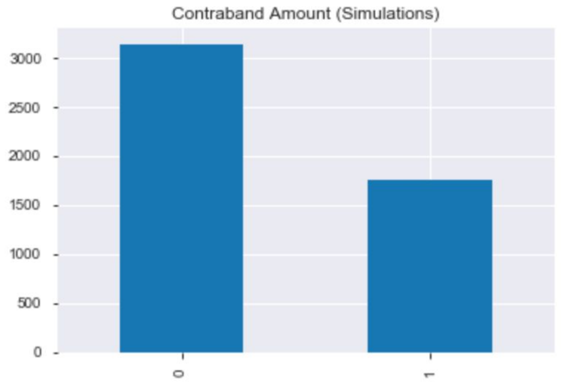
Regarding fairness, our model kept a discrepancy less than 5% between genders, but it was biased against White people. However, the model received twofold observations about White Race. Our model was not able to identify contraband from any Asians or Indians, but predictions were more precise as to Whites and two times less precise to Black. Also, our model performed more accurately predicting the occurrence of contraband for women than for men. Concerning ethnicity, our model predicted contraband for 19% of Hispanics who actually had contraband and 37% for non-applicable. Estimation of fairness for predictions to all received observations will be considered in Fairness chapter.

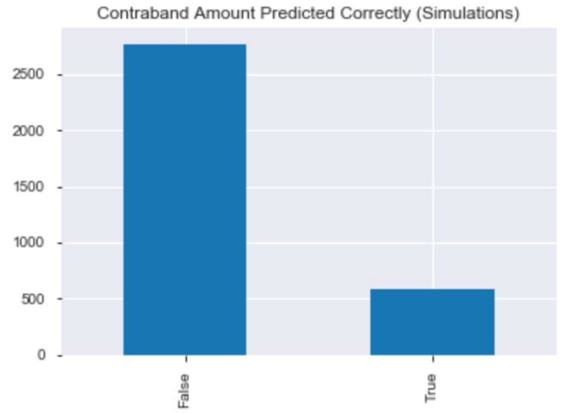
As it was previously uncovered in the first report, our model was built on predictions whereby contraband success rate has been equal 37%. This might mean that to overcome this result, we would need more data with the success rate at least 50%. The actual recall rate for received simulations is close enough to contraband rate for trained data, but so far, indeed, far apart from expected. The further possible solution will be to overview the simulations and predictions on them, to compare the results with the trained dataset and then to retrain our model inclusively with the new data.

Result Analysis

Model Performance

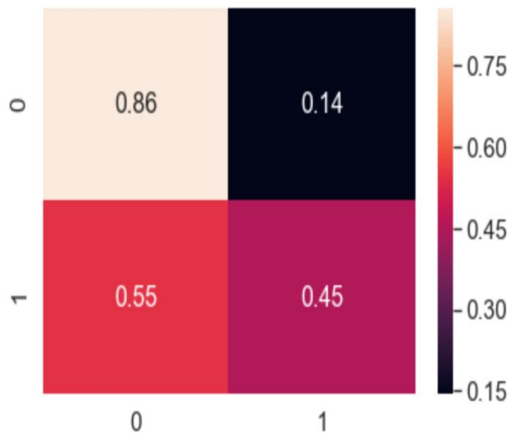
In this chapter we'll take a closer look at our model performance considering various metrics. Firstly, let's check the amount of predicted contraband regarding provided simulations:



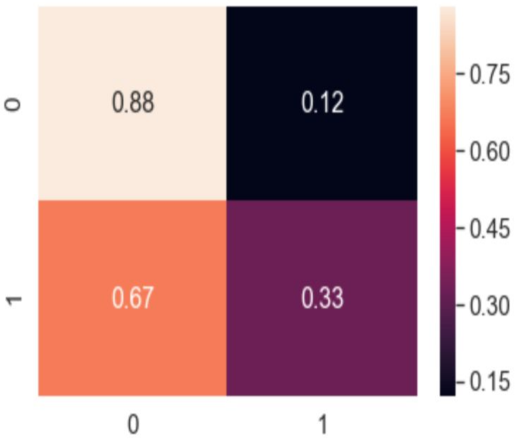


Therefore, from 1761 contraband events, the contraband rate from our simulations equals 36%, however, we were able to predict correctly 579 events, i.e., 17% contraband rate (579/1769 = 33% true positives).

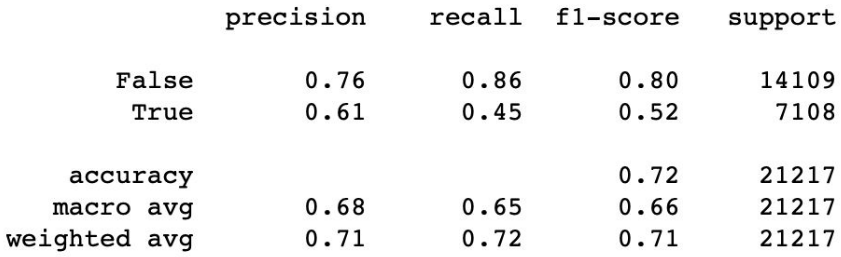
Expected confusion matrix:



Confusion matrix for simulations:

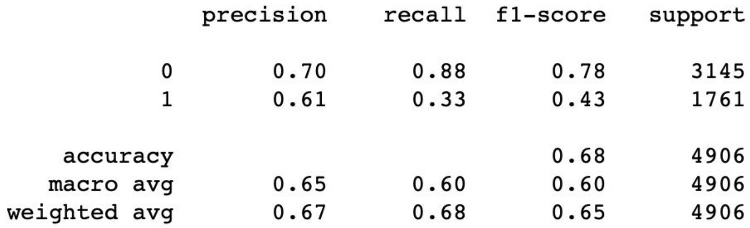


And respectively expected results from classification report for our test dataset:



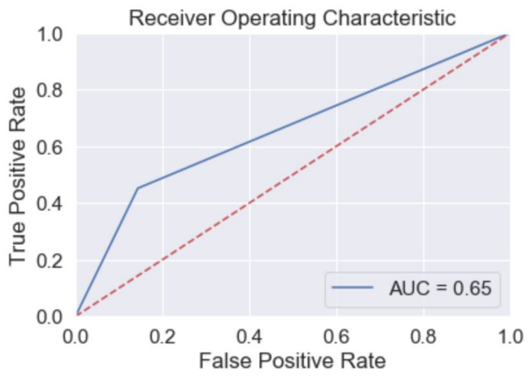
```
              precision    recall  f1-score   support

       False       0.76      0.86      0.80     14109
        True       0.61      0.45      0.52      7108

    accuracy                           0.72     21217
   macro avg       0.68      0.65      0.66     21217
weighted avg       0.71      0.72      0.71     21217
```

And for received simulations:

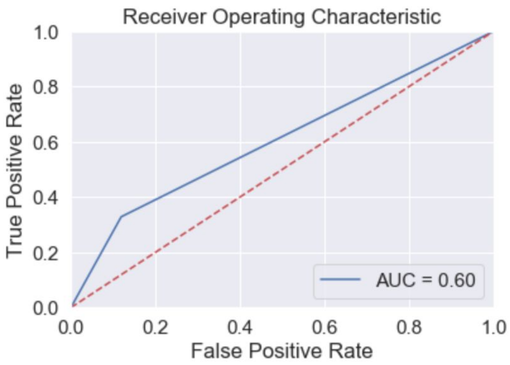


```
              precision    recall  f1-score   support

           0       0.70      0.88      0.78      3145
           1       0.61      0.33      0.43      1761

    accuracy                           0.68      4906
   macro avg       0.65      0.60      0.60      4906
weighted avg       0.67      0.68      0.65      4906
```

Expected:



In simulations:

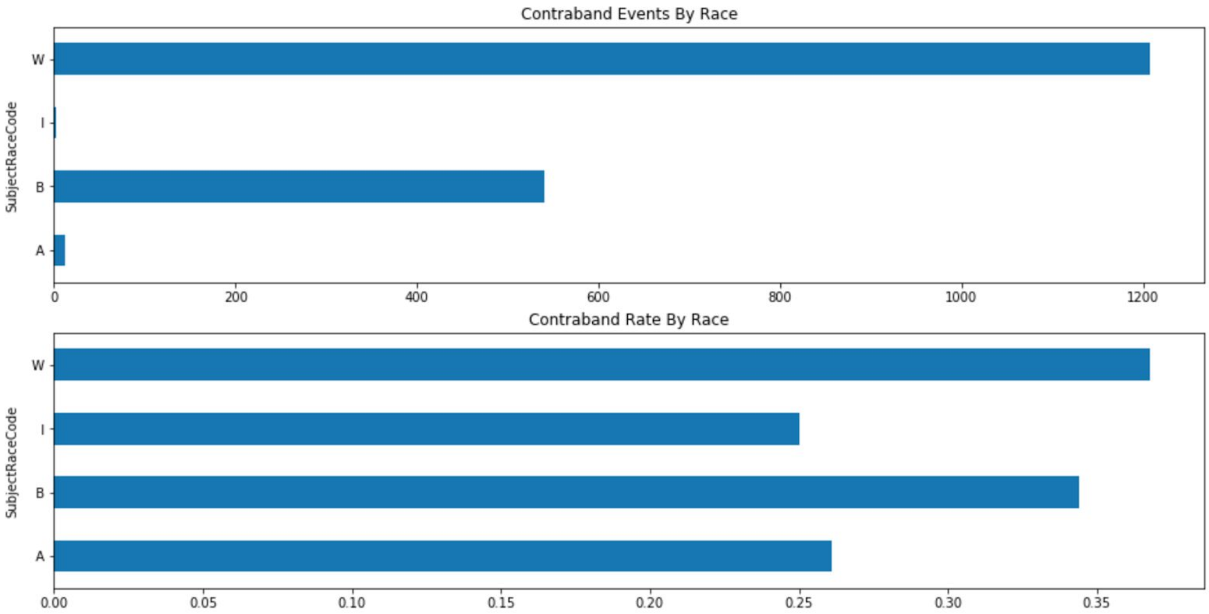


It’s crucial to get the point why the performance of our model is worse than expected, and the answer might lie on the provided sample. We’ll compare the simulations hereinafter with training dataset in greater detail.

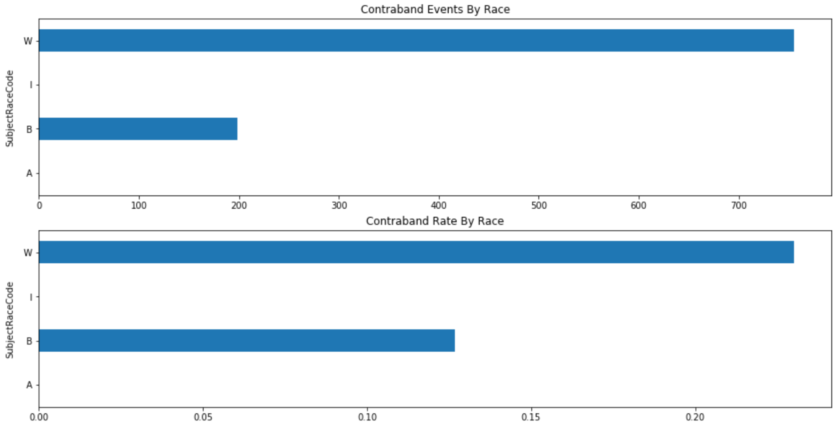
Fairness

The predicted contraband rate for all observations is 0.20, i.e. 1936 events from 9900 have been predicted with contraband presence.
In this chapter, we aimed to demonstrate the data within incoming simulations where the vehicles were searched (true class is not null), along with the predictions of our model for that searches and predictions for all received observations with null class inclusively, along with data we used to test our model for comparing the performance of our model both for simulations and test dataset.

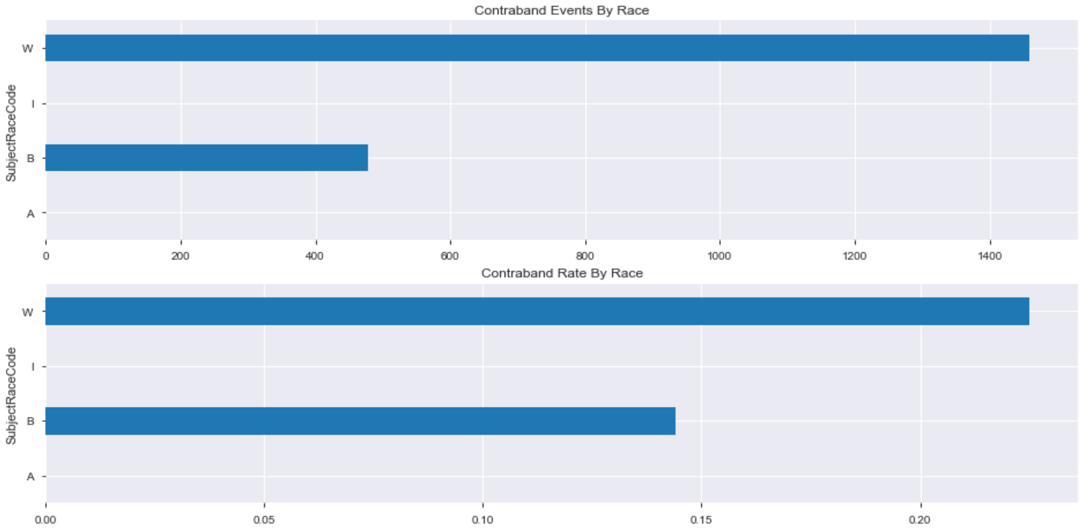
Regarding race we received the following data in simulations:



Let’s check out the predictions of our model for the observations with not null class:



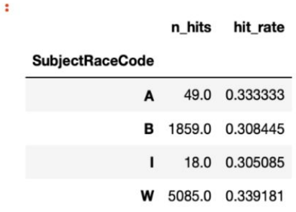
And below are predictions of our model for all received observations regarding race:



For all sent observations, the hit rate for Black equals 0.14 , for Whites 0.22. Still, within a small increase amount of vehicles for Asians and Indians (all simulated observations had 72 Asians and 32 Indians) our model was not able to identify any of them as non innocent.
Below expected performance based on the test set:

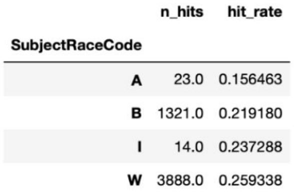
Original test set (race): (race):

Predictions of our model based on the test set



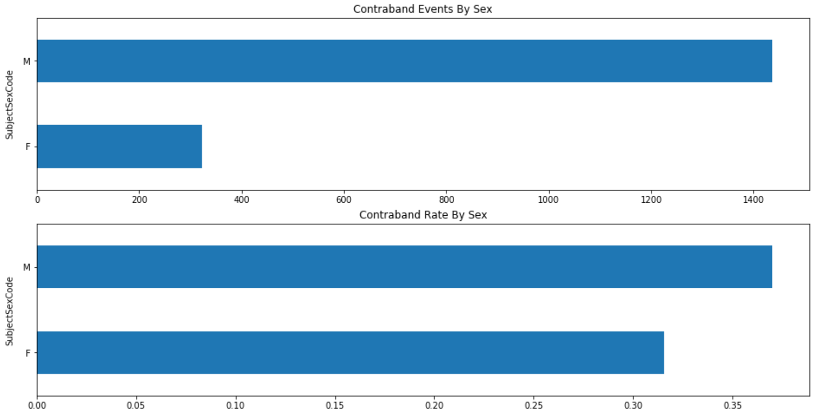
:

	n_hits	hit_rate
SubjectRaceCode		
A	49.0	0.333333
B	1859.0	0.308445
I	18.0	0.305085
W	5085.0	0.339181

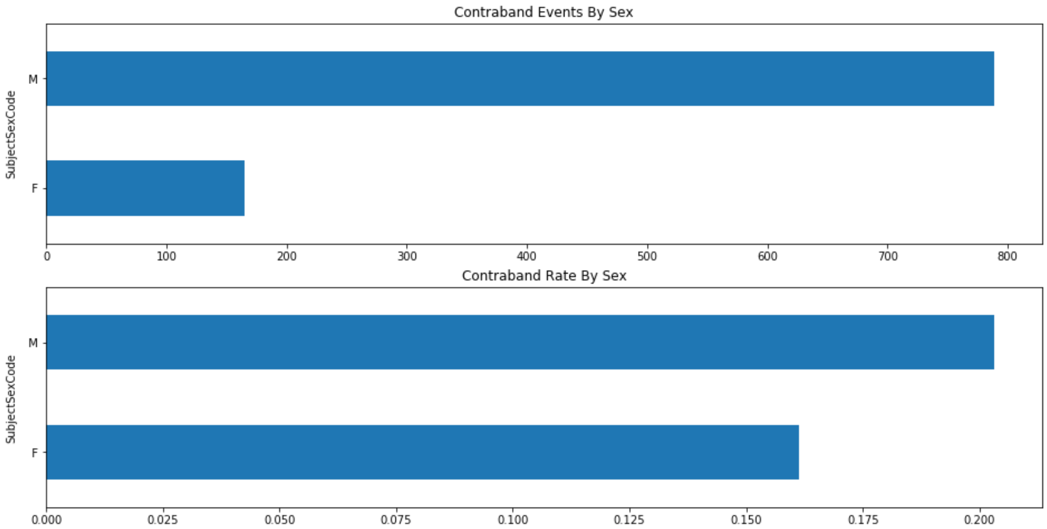


	n_hits	hit_rate
SubjectRaceCode		
A	23.0	0.156463
B	1321.0	0.219180
I	14.0	0.237288
W	3888.0	0.259338

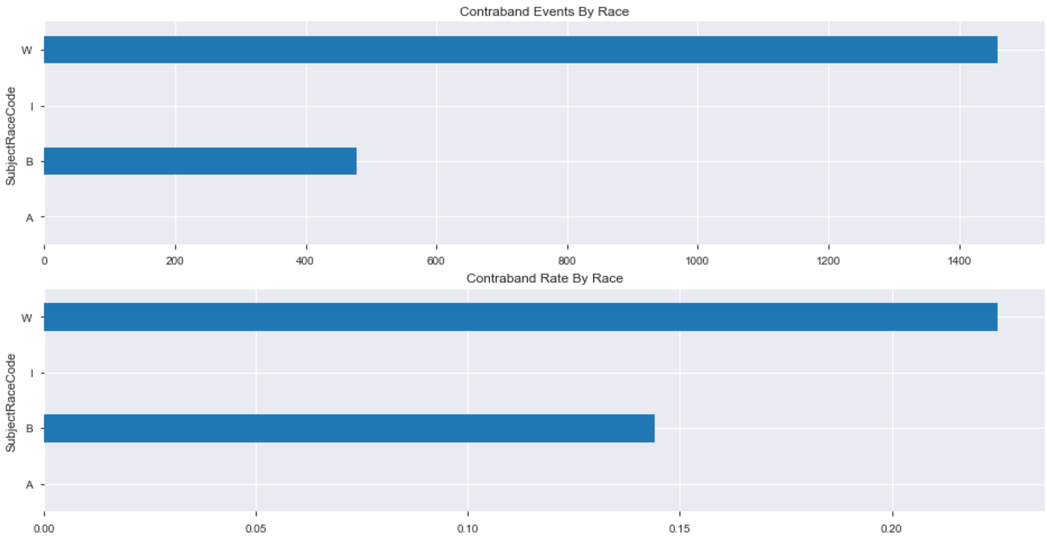
Regarding the gender, we received the following observations for searched vehicles:



For the searched vehicles our model predicted the following outcome for each gender:



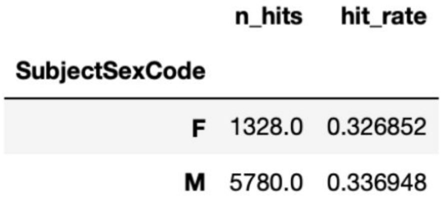
And to evaluate how fair our model is, let’s take a closer look at predictions for all received observations:



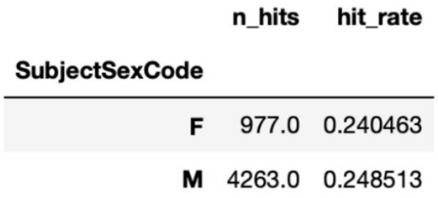
It turns out that the hit rate for women is 0.17 and for men is 0.20. Within all received observations, our model was able to keep the score no less than 5% between genders, therefore we can say it performed well regarding fairness criteria for genders.

And to regard expected performance based on the test set:

Original test set (gender): **Predictions of our model based on test set(gender):**

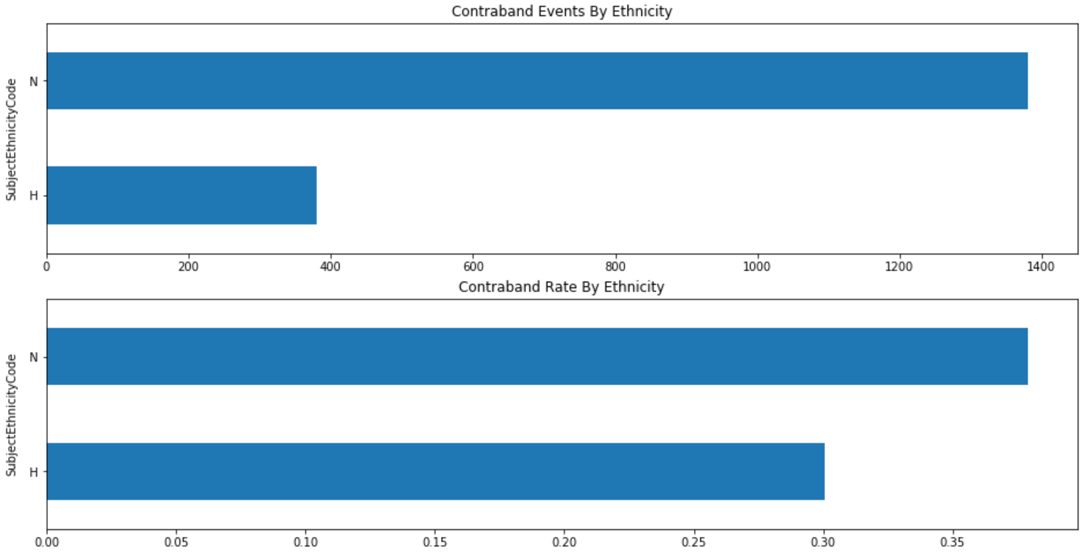


SubjectSexCode	n_hits	hit_rate
F	1328.0	0.326852
M	5780.0	0.336948

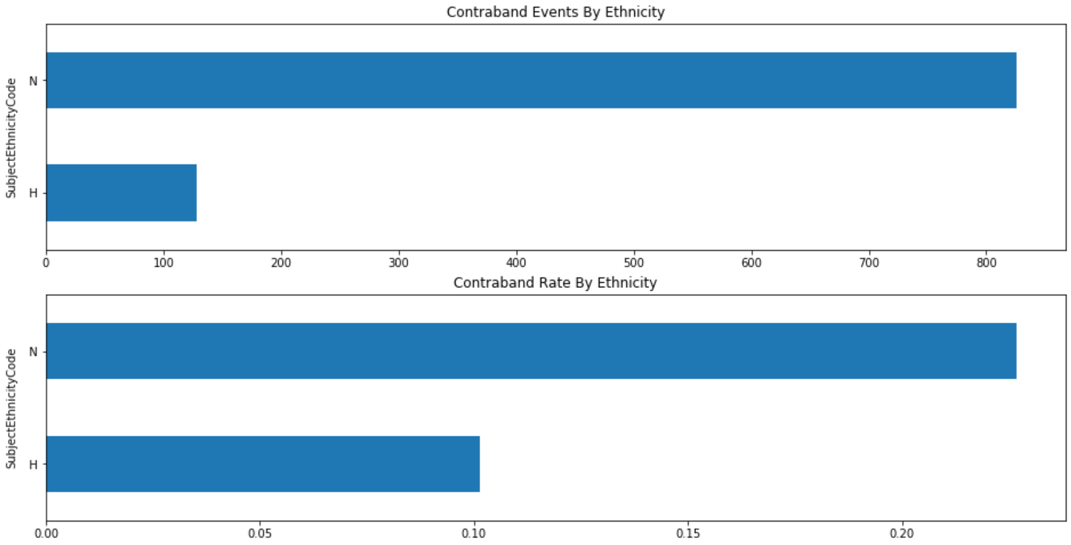


SubjectSexCode	n_hits	hit_rate
F	977.0	0.240463
M	4263.0	0.248513

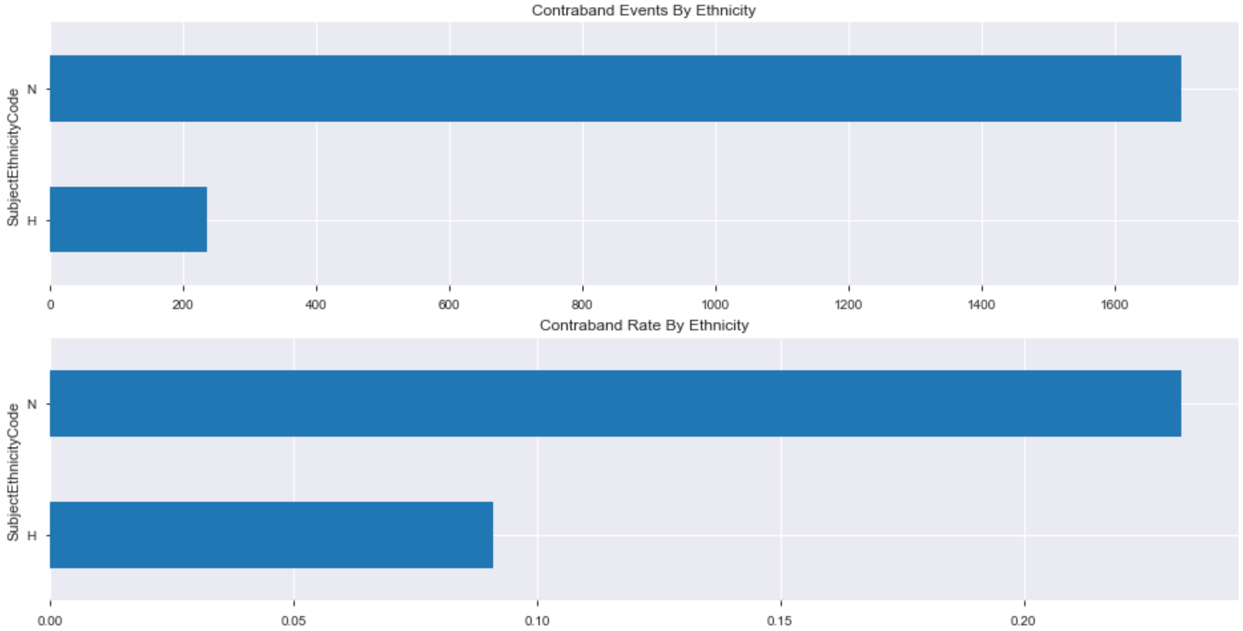
For ethnicity we received just one representative of Ethnicity class (Hispanic) and others were not applicable. In view of that, we received the following simulations:



For given not null class observations, the predictions were:

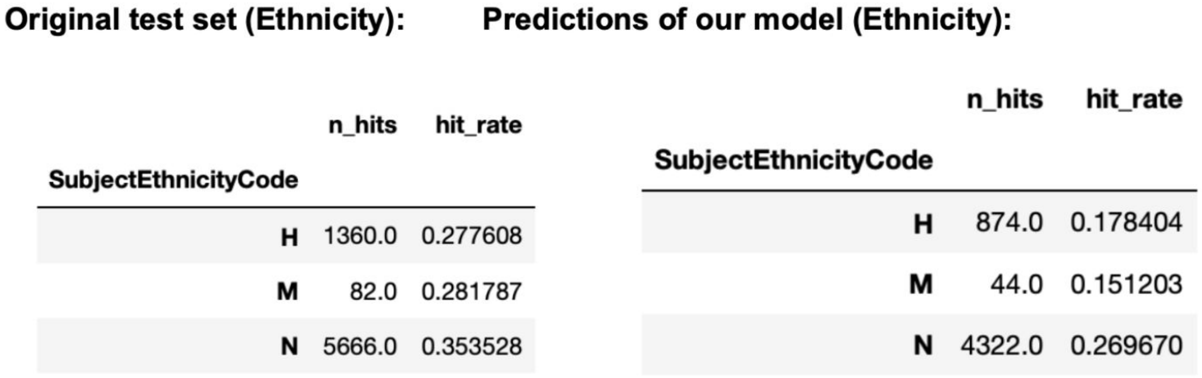


For all observations:



The rate for hispanic ethnicity was predicted as 0.09 and others as 0.23. Even though we were not able to keep a gap less than 5% between Hispanics and 'non applicable', we suppose it's a matter of sample.

Expected results from the test set:



Original test set (Ethnicity):

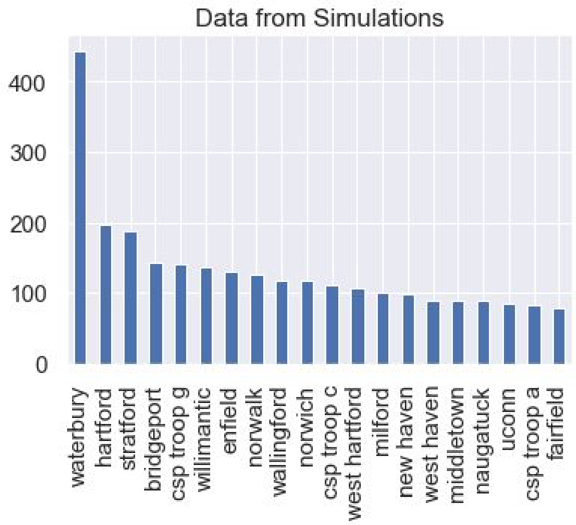
SubjectEthnicityCode	n_hits	hit_rate
H	1360.0	0.277608
M	82.0	0.281787
N	5666.0	0.353528

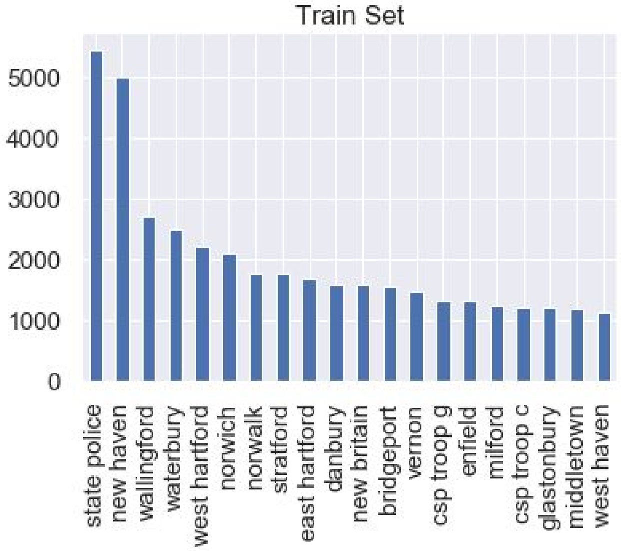
Predictions of our model (Ethnicity):

SubjectEthnicityCode	n_hits	hit_rate
H	874.0	0.178404
M	44.0	0.151203
N	4322.0	0.269670

Population Analysis

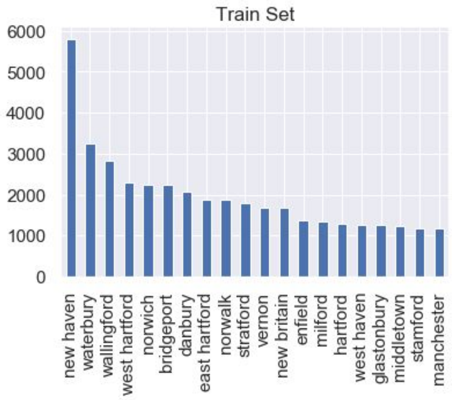
Although in this chapter the goal is to observe how population changed from the training set to simulations, we found that our app received a lot of data which is different from trained data regarding departments:

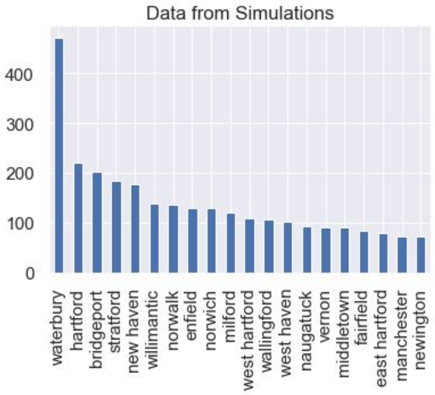




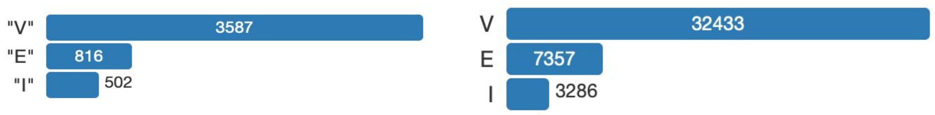
There was no state police in simulations, nonetheless 'new haven' department was presented in 99 of departments.

Regarding location:

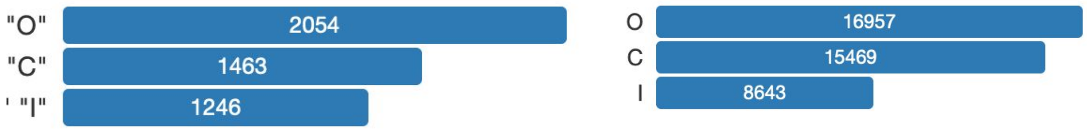




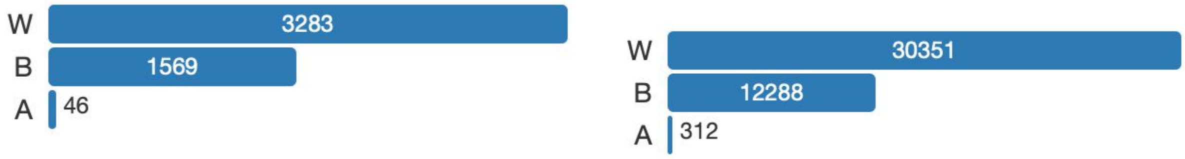
Intervention Reason received in simulations and for training has barely changed:



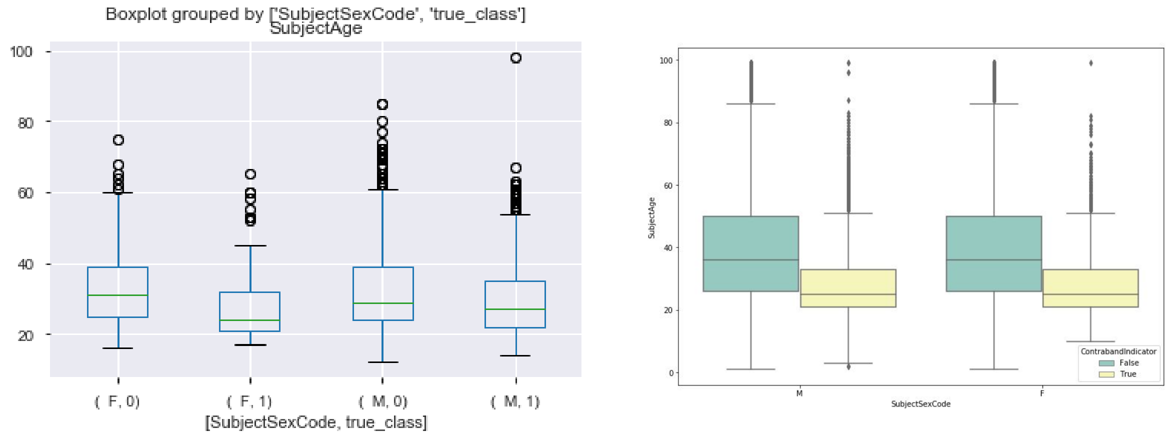
Search Authorization Code:



Frequency distribution of race in simulations and train set:



And regarding the age, sex and contraband rate:

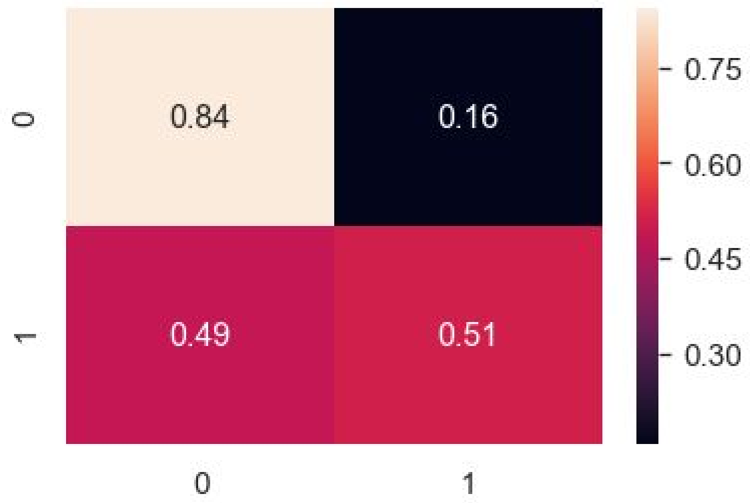


We found that population itself barely changed except of departments and ethnicity (as it was represented in a previous chapter, Middle Eastern ethnicity was absent in the received simulations).

There is almost no difference between Resident Indicator in the training set and simulations: 91% of residents and 9% of non residents in both datasets. The drivers were town residents in 40% of events in a training dataset, and in 45% in simulations.

Next Steps

Let’s join the received simulations and train data in the first instance to see if the new data facilitated our model performance.



Indeed, with the new information added, we’ve been able to increase recall on 6%, which equals 51% search success rate. However, it is important to emphasize that these results for the given dataset and at the current moment of production, we cannot ensure that our model will achieve the same success rate for real case scenario.
With the new data we also were able to keep no less than 5% discrepancy for protected classes.
The new information helped with improving the performance of our model regarding new test set, but as it was said in the previous report, data might be changeable over time and the received simulations represent that perfectly.
Since we are using Logistic Regression which gives us a probability, for keeping a high recall we can test the decrease of default threshold whether to search a vehicle, i.e. ‘=> 0.50’ is ‘True’ to ‘=> 0.40’ is True , but in that case the precision may suffer, so it should be discussed with the client as a trade-off.

Deployment Issues

Re-deployment

We re-deployed our model after dry run since the model was giving incorrect outcome, i.e. probability instead of a boolean response. After dry run we had some issues with re-deployment. With the new additional sample and corresponding retraining of our model, we are planning to re-deploy our model in the future.

Unexpected problems

Before the dry run we were not able to get a response from our application in a command line. The provided solution was a resetting of the database, which worked perfectly well. After the dry run, we realized that our app was giving a response as a probability, so after changing it to Bool, it was not working at all, particularly it was not dealing well with receiving repeated entries. The reset and restart didn't help either. The issue was solved after setting the right array in our probability response and converting it to a string respectively.

However, the most important problem happened with not receiving the true class for half of the received observations (5,000), i.e. the true class is null for them. The issue with receiving of true class occured after the 4999th observation, so we can not identify it with the initial problem. Nonetheless, our application provided with predictions all received observations.

What would you do differently next time

Due to the project objective, i.e. to keep a balance between 0.50 success rate along with requirement on fairness, our further goal will be training another algorithms for classification tasks and applying fairness packages.

As for the project we must test all possible algorithms with different configurations along with hyperparameters tuning, considering their computational power along with their complexity, explainability, and ease of implementation. So far the first most promising algorithm for the further testing is Xgboost. It has various tuning parameters for tree-based learners, therefore different configurations which potentially can significantly help to improve performance. Also Xgboost works very fast regarding large datasets, and as for the future, our database will increase, we will need something more stable and efficient on a long run. Xgboost is more sophisticated than logistic regression and it's more friendly in production than Random Forest Classifier which we as well considered before.

With getting a high performance, we might sacrifice fairness, hence for measuring bias against protected classes, we are going to try out aequitas, an open source bias and fairness audit toolkit that is easy to use as an addition to the machine learning workflow, enabling to test models for several bias and fairness metrics in relation to multiple population sub-groups.