To Do:

- Deal with the second requirement!

- Improve fairness definition.

- Deal with grouping categorical variables.

- Decide whether to make groups inside local, departm and stop\_reason.

- The loss function and its relation with classification threshold, precision, and recall.

- GridSearchCV with the appropriate scoring function.

- Dealing with missing values (output -1 and -2 from encoders)

- confidence intervals in bar plots

How to decide whether the success rate associated with a given feature’s category is meaningful, i.e., whether this category being present really changes the probability of success?

In the example above we have grouped observations by all the different locations where searches have taken place and plotted the histogram of the success rate. We see that some locations have a success rate higher than average (on the right of the red line) and other locations have a success rate below average. Some locations even have a success rate of 100% and others oh 0%. In order to decide whether the observed success rate is meaningful or is just pure chance, we perform a binomial test with the null hypothesis being.

Some discrimination is acceptable (earnings when asking for a loan), some is not (race)! Some is accepted in some contexts (age in health insurance) but not in others (age in job applications).

Features may be less informative or reliably collected for minority group(s); Biased population as an effect of the intervention (pg. 12);

1. Individual fairness: Are individuals treated by a decision-making system consistently independently of the social salient groups they belong to?

2. Group fairness: Do the outcomes of a decision making system systematically differ between social salient groups?

Parity (or equality) in impact (or demographic parity) (pg. 24) - Parity without ground truth (we don’t have the false negatives, drivers who were not searched but actually had contraband)

Impossibility Results & Trade-offs: 1) Between different measures of group fairness 2) Between fairness and accuracy (pg 40)

# Client requirements

## Summary

In this report, we present the results of a detailed statistical analysis on historical data, collected by the police department, concerning its traffic stop and search for contraband operations. The aim was to evaluate the existence of a bias against people of certain backgrounds in the search decision pattern used by police officers.

Additionally, we were asked to develop a model to objectively predict the risk of contraband based on the historical data provided. This model should be able to improve current search success rate while respecting acceptable levels of bias between personal classes (race, ethnicity and gender). A description of the proposed model and its expected results are presented in this report, the model itself has been deployed on an online platform[[1]](#footnote-0).

## Requirements clarifications

*Awkward Problem Solutions™* has been asked to comply with the following requirements:

1. A minimum 50% success rate for searches (when a car is searched, it should be at least 50% likely that contraband is found)
2. No police department should have a discrepancy bigger than 5 percentage points between the search success rate within protected classes (of race, ethnicity, gender)
3. The largest possible amount of contraband found, given the constraints above.

The reason for the third requirement is self-evident. If we can increase the percentage of contraband found we also increase the certainty of punishment, producing a stronger deterrent effect which prevents more people from committing contraband crimes. From a specification perspective, this is equivalent to ask for a maximization of the model’s recall score: for a given number of stops the existing contraband is either searched and found (true positives) or not searched and not found (false negative), and we want to find as much contraband as possible (true positives) given its total amount (true positives + false negative). This requirement provides us with a metric, or scoring, for the evaluation of the classification algorithm, whereas the first two requirements give us constraints that should be complied with.

Regarding the first requirement, it possibly arises from the existence of a cost associated with the searching procedure and a benefit associated with finding contraband. In that case, the rationale for this requirement should be the fact that the benefit resulting from finding contraband (let’s call it 2*s*) is valued as twice the cost of searching for it (call it *s*), on average. In this way, since every time a search is made we incur in the cost *s*, searching is ‘cost-effective’ only if the probability of obtaining the expected benefit *2s* by finding contraband is, at least, 50%. Once we have estimated the probability of finding contraband for a given observation, this requirement defines the classification threshold to be used by the classifier algorithm[[2]](#footnote-1), i.e. we will predict contraband and clear the search authorization only if we expect contraband to be found with a probability higher than 50%. The definition of the loss function (the cost and benefit associated with each possible decision) is a client’s exclusive responsibility and it is a central part of any classification algorithm, therefore we will stick to this requirement.

The second requirement addresses a concern with the model’s fairness among protected classes, i.e., the model should perform similarly regardless of the individual classes a subject belongs to. From a statistical perspective, we interpret the search success rate (SR) as a precision score requirement: among the predicted positives (search ‘clearance’), success is measured by the ratio number of findings (true positive) / number of searches performed (true positive + false positives). We certainly approve this fairness concern, but we can anticipate that we have not been able to ensure its compliance, neither at police department level nor at a global level. We have assessed this metric, both for the police officers performance and for our model’s expected performance, but we have not been able to comply with it. Even so, we have suggested improvements to the model that should allow us to deal with this requirement at a future time.

# Dataset analysis

## General analysis

To perform the requested analysis we were provided with a dataset concerning car stops, including whether the car was searched, and if any contraband was found. This dataset comprises records of 2,473,643 car stops between October 2013 and May 2018 and is described by 16 variables. A brief characterization of the dataset is presented below.

* *VehicleSearchedIndicator*: ‘True’/’False’ variable indicating whether the vehicle was searched. We have 76,743 searches in a total of 2.47 million stops, i.e. only about 3,1% of stopped cars are searched.
* *ContrabandIndicator*: ‘True’/’False’ variable indicating whether contraband and/or evidence was discovered. True for 28,341 observations, which amounts to 1.15% of total and 33.25% among searched vehicles.
* *Department Name*: There are 122 police departments in the dataset. The number of traffic stops per department is very dispersed. While the ‘State Department’ accounts for 13% of observations, there are 12 departments with less than 1,000 observations and, between these, 6 departments with less than 100. The average value of traffic stops per department is 20,275 and the 50th percentile is 14,754 stops.
* *InterventionLocationName*: Location of the intervention, 1,500 places reported. Since each police department may perform traffic stops at different places, in this case, the dispersion in the number of traffic stops per local is even larger than above. We have more than 50% of the locations with a single stop reported; 76% locations with 5 or less stops; while for those 75 locations above the 95th percentile the average number of stops is 26,254.
* *InterventionDateTime[[3]](#footnote-2)*: Date and time of the intervention, ranging from October 2013 and May 2018, with a break between April 2015 and September 2015.
* *InterventionReasonCode*: Code for the reason given for stopping the vehicle. Three distinct values: ‘violation’ (88% of the stops, i.e. 2,179,595 observations); ‘equipment’ (10%); ‘investigation’ (2%).
* *StatuteReason*: Reason given for stopping the car. Fifteen distinct classifications. Most frequent are: ‘speed related’ (in 27.5% of stops); ‘defective lights’ (9.2%); ‘registration’ (9.2%); ‘cell phone’ (9.0%); ‘moving violation’ (7.7%); and ‘traffic control signal’ (7.2%). ‘other’/’other error’/missing amount for 8.9%.
* *SearchAuthorizationCode*: Authority to search the vehicle. Three distinct values: ‘Consent’ (35.2% of searches); ‘Inventory’ (20.5%); ‘Other’ (40.0%), which encompasses probable cause, reasonable suspicion, plain view contraband, incident to arrest, drug dog alert or exigent circumstances. Missing values or ‘Not Applicable’ in 4.3% of the searches.
* *ReportingOfficerIdentificationID*: There are 8,593 distinct officers in the dataset.
* *ResidentIndicator*: ‘True’/’False’ variable indicating whether the subject was a resident of the state. In the complete set, 86% of drivers stopped are state residents, whereas considering only the searched vehicles 91% are from state residents.
* *TownResidentIndicator*: ‘True’/’False’ variable indicating whether the subject was a resident of the town. In the complete set, 69% of drivers stopped are town residents, whereas considering only the searched vehicles 60% are from town residents.
* *SubjectAge[[4]](#footnote-3)*: Age of the main occupier of vehicle. Average age is 39 while median is 36.
* *SubjectEthnicityCode*: Officer perception of the ethnicity of subject. Two distinct values: ‘Hispanic’ (13.3% of all subjects, i.e., 328,450) and ‘Middle Eastern’ (1.8% or 45,561 subjects). ‘Not Applicable’ accounts for 85% of observations.
* *SubjectRaceCode*: Officer perception of the race of subject. Four distinct values: ‘White’ (81.6% of subjects), ‘Black’ (15.6%), ‘Asian/Pacific Islander’ (2.0% ) and ‘Indian America/Alaskan Native’ (0.8%). No missing or unknown data, all observations belong to one of the four classifications.
* *SubjectSexCode*: Subject’s gender. Two distinct values: ‘Male’ (63.2% of subjects) and ‘Female’ (36.8%). No missing or unspecified sex.

**Unexpected observations or missing values**

* *ContrabandIndicator*: Expected to be ‘False’ when no search was performed (i.e, *VehicleSearchedIndicator* equal to ‘False’), but there were 2,823 observations with contraband finding without search indication. For training the model these were ignored since we considered only those observations with VehicleSearchedIndicator equal to ‘True’.
* *SearchAuthorizationCode*: Expected to be missing or ‘Not Applicable’ when no search was performed (i.e, *VehicleSearchedIndicator* equal to ‘False’). However, 15,360 of stops without search indication have a valid *SearchAuthorizationCode* classification.
* *TownResidentIndicator*: All state residents were expected to be town residents as well. Nevertheless, 44,904 cases exist (1.82% of the dataset) where town residents are not state residents.
* *SubjectAge*: 1,117 unexpected observations with subject’s age below 16 years old (4.8%). Also no subject older than 99 years old. It is likely that all subjects with more than 100 or more years of age have been classified as being 99 because the number of subjects with this age is 661, while the number of subjects with the age of 98 is 132 and with the age of 97 is 216. Another possible justification for the unexpected ages, is that ages were computed from the date of birth and the year of birth was recorded with only two digits.

## Business questions analysis

The first concern manifested in the briefing was the suspicion that the search decision criterion used by police officers might be biased against people of certain backgrounds. Considering being searched as a sample process from the complete population (in our case, drivers who are pulled over), bias happens if individuals are not equally likely to be selected, particularly if the difference in the probability of being searched arises from belonging to a certain class (race, ethnicity or gender). Under this scenario, certain population classes will be underrepresented or overrepresented in the sample (i.e. in the searches). The plots presented below show us that this effect is indeed happening. For example, ‘Black’ subjects are overrepresented in searches when compared to all traffic stops, since the presence of this race class becomes higher when we consider only searched drivers instead of all drivers. The same occurs with ‘Hispanic’ class and ‘Male’ class.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Having said that, we must remember that the purpose of a vehicle search is not to draw a random sample from the population, it is to find contraband or crime evidence, and therefore it may be reasonable to accept that the search criterion favours (in other words, biases) certain classes over others (as well as other characteristics, such as driver’s age or driving during certain hours of the day). In this case the rationale is the following: if we know that the presence of certain characteristics makes contraband or crime evidence more likely to be found (i.e., if these characteristics are correlated with evidence finding), then we may use this information to increase the search success rate. The search decision will be biased, but it may be fair. This behavior is even desirable if we want to increase certain metrics, as requested in the briefing.

Then, accepting a bias in the search criterion might be fair and justifiable if the evidence shows us that search findings are actually more likely in those classes we have bias against. However, what the plots below show us is that discrimination was either not justifiable, or was taken to a level in which it became unjustifiable. Except for the case of gender, the overrepresented groups in the searched subset, ‘Black’ and ‘Hispanic’, actually have a lower finding success when compared to other groups, as ‘White’ or ethnicity ‘Not Applicable’. This results indicate that the current search criterion is unfair.

A first step in the direction of a fair classification model is to remove the protected classes from the dataset. Although it may not be a sufficient measure, it is a significant one.

|  |  |  |
| --- | --- | --- |
|  |  |  |

We will now focus on the model for contraband detection. First of all, we investigated the predictive power of each variable for contraband detection, in other words, whether or not the probability of finding contraband changes when a given variable changes its value.

Are values statistically significant? So far we didn’t mention this because we have been dealing with large numbers, but with the office departments we have to.

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |

Regarding all the other features that have not been used. The features which identify protected classes (*SubjectRaceCode*, *SubjectEthnicityCode*, and *SubjectSexCode*) we have already justified its exclusion as a first attempt to have a fair model. Besides, when they were included in the final model, just with the intention of comparing the results, the differences in the score metrics were completely negligible. For the reporting officer’s ID, even if we believe that certain officers have developed a kind of ability to detect the presence of contraband (which is unlikely: the different success rates between officers, even when statistically significant, are probably more related to the department and location they work at), since our purpose is precisely to replace their decision, it becomes useless to use this information. Besides, once the model is deployed, the information that some officer used to have a high success rate is irrelevant because we are not going to be provided with his perception of the situations’ risk, only with his ID. For the *InterventionLocationName*, to some extent it reproduces the *Department Name* information. Among the 1,500 different locations present in the dataset, 86% of them are serviced by a single department. This means that using *InterventionLocationName* together with *Department Name* is somehow redundant. Even though we will lose some information, we have opted for the last feature because it has a lower number of distinct values (122 against 1500). *ResidentIndicator* and *TownResidentIndicator* appear to have some predictive power, namely a slight increase in the success rate when any of these features takes the value ‘False’, yet when included in the model its performance doesn’t change, and therefore were rejected. A very similar case happened with *InterventionReasonCode*.

Explain that departments with few stops have more uncertainty about success rate. The same is true for the location (but even worse) and probably carries the same information as the department, that’s why we don’t use location.

## Conclusions and Recommendations

Some variables clearly enhance the predictive

# Modeling

## Model expected outcomes overview

Expect success rate among searched cars (precision) around 60%. Expect to find around 50% of existing contraband (or to miss 50 out of 100 existing contraband). Although we don’t have imposed any limit, expect discrepancies between protected classes to be low.

Compare to a random classifier as a baseline.

Dynamic evaluation is not useful in this case because the first requirement determines a minimum threshold of 50%, while the third requirement imposes a score metric that decreases with the increasing of the threshold. The combination of the two requirements lead us to define a threshold of 50% and to perform the evaluation of the model at this static position.

## Model specifications

In order to develop the model we have split the dataset into a training set, with 75% of the observations, and a test set, with the remaining observations. Five features from the original dataset have been selected: *Department Name*, *SearchAuthorizationCode*, *StatuteReason*, *SubjectAge* and *InterventionDateTime*. Below, we describe the pre-processing implemented in each of the features.

Explain the exact pipeline.

“and data cleaning and hyper-parameter decisions are highly relevant.”

* *Department Name*, *SearchAuthorizationCode*, *StatuteReason*
  + Text cleaning: Non-letter characters are replaced by spaces; multiple spaces are replaced by a single space; leading and trailing spaces removed; string converted to lower-case (implemented with a user developed class).
  + Ordinal encoding: since we are dealing with categorical text data and scikit-learn classification doesn’t accept text, we had to convert it to numerical values.
* *SubjectAge*
  + Binarization: In order to reduce the number of existent values a binarization was used. Bin limits span from 21 until 66 years old, with a constant width of 5 years.
* *InterventionDateTime*
  + Hour of the day: From date and time we have selected only the hour of the day.
  + Binarization: Classification as low finding rate hours (from 2 a.m. to 11 a.m. together with 4 and 5 p.m.) and high finding rate hours (remaining hours)

After, the model was pickled has a way to persist for future use in the deployment part without having to retrain it.

‘Clearance’/’No clearance’

## Analysis of expected outcomes based on the training set

In-sample-error (ISE) or training error vs. Out-of-sample error (OSE) or testing error

Confusion matrix for the complete set and for each protected class separetly;

Plots ROC and FPR/TPR against threshold.

Define a baseline and compare with: historical data; random classifier.

## Alternatives considered

We have tried two main approaches in the development of the classification model. The first and simpler approach, since we were dealing mainly with categorical variables (with the exception of *SubjectAge* and *InterventionDateTime*), some even with a high cardinality like *Department Name*, was to leave the variables more or less in their natural state and use a classifier that could deal with them, like the Random Forest Classifier. This ended up being our most successful model and the option we have implemented. It was presented in the previous sections.

The approach described above is not adequate to be implemented with some algorithms, like the Logistic Regression, that are more suited for numerical variables (i.e., in which the numbers have a quantitative meaning and are not exclusively a label attributed to the observation). In order to use those kind of algorithms with categorical variables, the most common solution is to apply a one-hot encoding to them. In our case, since we have used *Department Name* (122 categories), *SearchAuthorizationCode* (3 categories) and *StatuteReason* (15 categories), this means we have replaced 3 features by 137.

A third approach, which we have not attempted, would be to find a meaningful way of converting the categorical variables into numerical ones. We have in mind, for example, mapping each category to one or more relevant values related with that category (e.g. replace the name of each police department by its average success rate in the training set), and then try to those algorithms suited for this kind of variable. In this case we had to do regularization and standardization of variables

One-hot encoding for all variables except age and then Lasso penalization or select the most meaningful features.

## Known issues and risks

Subjects may adapt to the model. For example, if for older people the chance of being searched is low, because the success rate among older people was low on the training set and the model learned that, then people might realize this characteristic of the model and start using it to their advantage. A simple way to mitigate this risk would be to include a random element in the ‘Clearance’/’No clearance’ decision (i.e., to give the officer a ‘Clearance’ instruction if the classifier’s predicted probability is greater than the threshold, or if a random variable decides so), such that even a person who is very knowledgeable about the classifier behaviour would never be sure of its output.

We could expect the model performance on real-world data to be the same as on the test set (see [Analysis of expected outcomes based on the training set](#ujiy8nh7d8i)) only if the whole process was to be maintained: a first selection of subjects by the police officers, and only then a search decision provided by the classifier. However, the aim of our model is to completely replace the police officer’s searching decision, since we want to get a prediction every time a car is stopped. The problem is that we have trained our model on a sub-sample of stopped cars, those which have been searched by police officers, and this sub-sample we had access to might not be an accurate representation of our population (stopped cars). If this happens to be the case, we are facing a problem of selection bias.

This effect might undermine the ability of our results to be generalized to the complete population.

“the distribution of the data used to build the model will be different from the data that it will make predictions on”

Option 1) “use a subsample of the unlabled data, that is the cases that were not investigated, and consider them as non-targets.” Jacobusse, Gert & Veenman, Cor. (2016). On Selection Bias with Imbalanced Classes. 325-340. 10.1007/978-3-319-46307-0\_21.

Option 2) “reweighting the samples according to the ratio of class samples”

Selection bias can take different forms, but in this case the problem is a “systematic error due to a non-random sample of a population, causing some members of the population to be less likely to be included than others, resulting in a biased sample”

If the training set is a random sample from the population, then the classifier predicted probabilities are expected to be unbiased. However, if we admit the possibility that there was some bias in the drawing for the training set, i.e. if the police search decision was itself biased, then the predicted probabilities will reflect this effect.

Possible solution: also random selection and then use the true classes of these random selected observations to retrain the classifier.

# Model Deployment

## Deployment specifications

We have integrated our model in a web application developed with Flask and deployed it to heroku, in order to make it online available[[5]](#footnote-4). The web application has two basic functionalities:

* Predict[[6]](#footnote-5): Returns a search ‘clearance’/’no clearance’ instruction for the specific observation received. It is the user's responsibility to provide the model with the observation’s data. A classifier, which has previously been trained and uploaded to the model, is fed with the observation and outputs a predicted probability of finding contraband. ‘Clearance’ is provided to the user if the predicted probability exceeds the 50% threshold (see [Requirements clarifications](#z1d5j97jevo9)), ‘no clearance’ is returned otherwise.
* Update[[7]](#footnote-6): The deployed model accepts user updates about the true class of previously sent observations. This functionality allows us to perform a later assessment to the model performance, and eventually retrain the classifier, using the received updates.

In order to keep a record of the model’s operation we have connected it to a database table having the following columns:

* Observation ID: Integer value which uniquely identifies the observation. It is the responsibility of the user requesting the predict service to provide this unique ID (together with the observation itself).
* Observation: Observation’s data, in json format, provided by the user in the predict request.
* Predicted probability: The probability of finding contraband, as predicted by the classifier running in the model.
* True class: The true class of the observation (*ContrabandIndicator* ‘True’/’False’). Available only if the update service has been provided by the user for the observation (not required), otherwise is ‘NULL’.

In order to make the predict and updated services operational, as well as to connect to the database, when we launch the model the following steps are performed one time:

* Create an instance of a PostgreSQL database (if running locally then an instance of a Sqlite database in the model’s folder is created instead). This instance is used to manage the connections to the database. We have used the peewee ORM.
* Define a class with the structure of the database as described in the previous paragraph.
* Load the trained classifier (a pipeline with the feature preprocessing and the classifier itself - as described in subsection [Model specifications](#6m6gr1ofi7v8) - has been previously fitted to the training set and serialized, using scikit-learn joblib’s dump function).
* Start running the web application, making the predict and update services available.

## Known issues and risks

* We have implemented no security measures whatsoever. Any person with knowledge of the application can access the predict and update services, populating the database with trash observations or replacing the true values provided by trustworthy updates.
* We currently have a limit of 10,000 observations in our database. It is reasonable to expect that this limit can easily be exceeded.
* The very simple interface to our web application, created with Flask, is really not user-friendly and will probably hinder the officers job.
* Using Flask micro-framework, while providing us with a quick and simple way to set up a HTTP server focusing on including only what we really need, lacks the structure and scalability provided by a standardized framework, like Django, with a large online community and extensive documentation.

# Annexes

## Dataset technical analysis

## Business questions technical support

## Model technical analysis

Decision tree example.

## Model deployment additional specifications

The observation must be provided in json format. It is recorded in the database ‘as received’. Although the features listed below are expected, they are not required

['Department Name', 'InterventionDateTime', 'InterventionLocationName', 'InterventionReasonCode', 'ReportingOfficerIdentificationID', 'ResidentIndicator', 'SearchAuthorizationCode', 'StatuteReason', 'SubjectAge', 'SubjectEthnicityCode', 'SubjectRaceCode', 'SubjectSexCode', 'TownResidentIndicator']

## Glossary of terms and formatting

In this section we present a definition of the terms and text formatting options that have been used in the report.

Italic: Feature names (e.g., *SubjectRaceCode*)

quotation marks: Different classes in a given feature (e.g., ‘Black’ or ‘White’ in *SubjectRaceCode*)

Finding(s):

Classifier: the process of predicting a class based on a received observation.

Model:

Class

Finding probability / probability of class 1 /

Search success rate (SR): percentage of findings

1. Available at https://heroku-app-model-deploy.herokuapp.com/ [↑](#footnote-ref-0)
2. Notice that the phrasing is not completely clear, as the sentence “a minimum 50% success rate for searches” may be interpreted as a request for an overall success rate of 50%. This interpretation would lead to constraining the precision score of the model to a minimum value of 50%. However, the clarification in parenthesis seems to indicate that the minimum 50% success rate should be attained in every single search, and not overall. We will follow this interpretation, which brings us to the option of defining a classification threshold at 50%. Notice the two concepts are closely related, since an increase of the threshold selected to classify each observation will also increase the overall success rate, and vice versa. [↑](#footnote-ref-1)
3. Plots of the monthly evolution of traffic stops and number of searches available in the annexes. [↑](#footnote-ref-2)
4. Histogram of subject’s age available in the annexes. [↑](#footnote-ref-3)
5. At https://heroku-app-model-deploy.herokuapp.com/ [↑](#footnote-ref-4)
6. More details provided in [Annexes - Model deployment additional specifications](#gi78qkf9gn2e). [↑](#footnote-ref-5)
7. More details provided in [Annexes - Model deployment additional specifications](#gi78qkf9gn2e). [↑](#footnote-ref-6)