# Sprint 1 documentation

The first sprint was designed to implement a base counterfactual explanation and qualitatively assess how that translates to the problem at hand. Another

1. What do experts look at when determining the NOC?

<description of workflow with input from corina, survey, literature>

1. What purpose does an explanation of the NOC machine learning model serve?

<see question 1, help understanding of the model>

1. What does the problem look like?

<see description of the data, the ML problem, data analysis results>

1. What type of explanation could work for this problem?

<see survey results, see local explanation of single data point options>

1. How can we make a counterfactual explanation for this problem?

<see counterfactual results>

Feedback received on the implementation:

Je laat nu 1 profiel zien als counterfactual; over het algemeen commmuniceert dit misschien de verkeerde dingen (e.g. TAC verhogen om van 3 naar 2 donoren te gaan). Het is voor dit probleem niet handig om meerdere counterfactuals te laten zien; mensen gaan daar niet naar kijken en extrapoleren wat de juiste randwaarden zijn. Wanneer je gaat simuleren, kun je misschien de verdeling van de feature veranderingen beter samenvatten.

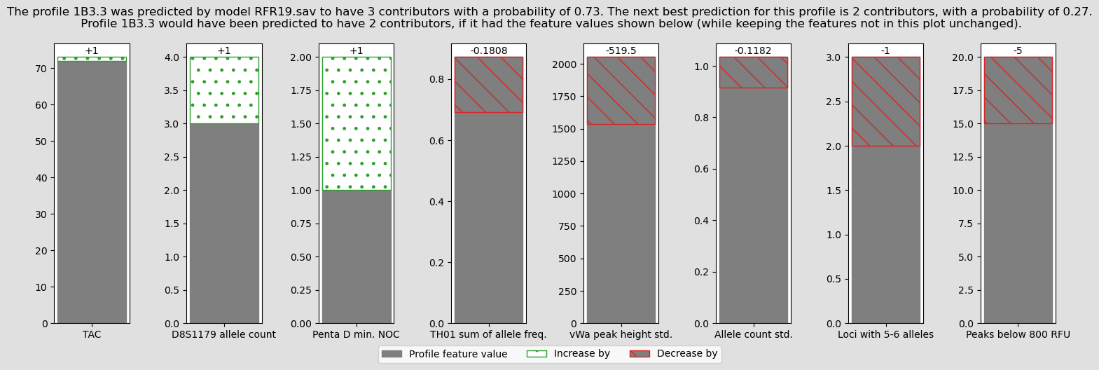
Misschien kun je ook een profiel laten zien ipv features.

Sprint 2

Factual and counterfactual explanations: anchored counterfactuals

Issues found with “normal” counterfactual methods:

* When generating counterfactuals based on the **training data**, the presented profile has particularities that do not represent how the machine learning model works in general. For instance, when investigating a profile which is predicted to have 3 contributors, we select the profile that is closest to our original profile in feature space that has 2 contributors. The differences between these profiles represent the feature changes that would have made the original profile to be predicted as having 2 contributors. However, since we are presenting just a single profile, we are essentially **overfitting** the counterfactual. An example can be seen in the picture below; It shows how increasing the TAC (total allele count), as well as an allele count at two specific loci would decrease the NOC from 3 to 2. This is **counterintuitive** since more alleles usually corresponds to more contributors.



Experiments with anchored counterfactuals:

1. Generate anchored counterfactuals by taking the closest CF data point and creating an anchor for it.

* Generate an anchor for the current data point
* Find the closest counterfactual data point
* Generate an anchor for the CF data point
* Compare and contrast the anchors

**Example**:

*Profile 1B3.3 was predicted by model RFC19.sav to have 3 contributors, with a probability of 0.45. The model will predict* ***3 contributors 96%*** *of the time when ALL the following rules are true:*

* *TAC <= 79.00*
* *Loci with 5-6 alleles > 1.00*
* *D8S1179 allele count <= 3.00*

*These rules apply to original data with a probability of* ***0.02***

*Profile 2.41 was predicted by model RFC19.sav to have 2 contributors, with a probability of 0.80. he model will predict* ***2 contributors 97%*** *of the time when ALL the following rules are true:*

* *Allele count std. <= 0.93*
* *Random match probability <= 0.00*
* *D8S1179 allele count <= 2.00*

*These rules apply to original data with a probability of* ***0.02***

**Idea**: If we change the original data point so that it matches the CF anchor as shown below:

* D8S1179 allele count 3.000000 -> 2.0
* Allele count std. 1.034381 -> 0.7
* Random match probability 0.000000 == 0.0

**We indeed get the counterfactual prediction: 2.0 with probability 0.67 while only having changed 2 features!**

**Problem**: original data point fits the rules of the cf data point OR when changing original data point to fit the CF anchor, the prediction stays the same.

*Profile 2.44 was predicted by model RFC19.sav to have 1 contributors, with a probability of 0.87. The model will predict* ***1 contributors 93%*** *of the time when ALL the following rules are true:*

* *TPOX min. NOC <= 1.00*
* *Allele count std. <= 0.65*

*These rules apply to original data with a probability of 0.29*

*Profile 1B2.3 was predicted by model RFC19.sav to have 2 contributors, with a probability of 0.91. The model will predict* ***2 contributors 100%*** *of the time when ALL the following rules are true:*

* *Allele count std. <= 0.83*
* *TAC <= 79.00*
* *vWa peak height std. > 3024.43*

*These rules apply to original data with a probability of 0.01*

Changing the features that do not align with the CF anchor in the original data point

* TAC 4.700000e+01 == 47 (<= 79)
* vWa peak height std. 1.080000e+02 -> 5000
* Allele count std. 4.642208e-01 == 0.464 (<= 0.83)

**Prediction: 1.0 with probability 0.87**

So even though the anchor holds for our data point, the prediction has not budged. Not even the probability.

Changing one of the features so that the **original anchor does not hold anymore**, does seem to change the prediction.

* Allele count std. 4.642208e-01 == 0.79 (<= 0.83, </= 0.65)

**Prediction: 2.0 with probability 0.48**

However, it apparently matters *how much* we change it.

* Allele count std. 4.642208e-01 == 0.70 (<= 0.83, </= 0.65)

**Prediction: 1.0 with probability 0.59**.

Perhaps this has to do with how anchors are *binned*.

Looking at feature Allele count std., the

* 0:'Allele count std. <= 0.38'
* 1:'0.38 < Allele count std. <= 0.45'
* 2:'0.45 < Allele count std. <= 0.65'
* 3:'0.65 < Allele count std. <= 0.83'
* 4:'0.83 < Allele count std. <= 0.93'
* 5:'0.93 < Allele count std. <= 1.05'
* 6:'1.05 < Allele count std. <= 1.17'
* 7:'1.17 < Allele count std. <= 1.28'
* 8:'1.28 < Allele count std. <= 1.44'
* 9:'Allele count std. > 1.44'