# *Igrannonica* – списак захтева апликације

*Igrannonica* је замишљена као игралиште за играње неуронским мрежама. То је веб апликација која треба да задовољи одређене обавезне, пожељне и опционе функционалне захтеве.

# Сегменти апликације и технологије

Апликација је подељена на три сегмента:

1. *Frontend*
2. *Backend*
3. Део за машинско учење

Обавезне технологије:

* *Angular Framework - frontend*
* *.NET – backend*
* Компонента која се бави обучавањем мреже и прорачунима је замишљена као микросервис и треба потпуно самостално да ради и без остатка апликације.

Савети:

* *TensorFlow* је препоручена библиотека за сегмент који се тиче машинског учења. Такође, постоји и *PyTorch* библиотека.
* *Onyx* и *H5* су неки од стандардних формата за чување неуронских мрежа

# Обавезни захтеви

Овде набројане ставке представљају обавезне функционалности које апликација треба да поседује. Треба имати у виду да што већу могућност корисник има приликом избора различитих параметара и метрика, то боље.

* Могућност учитавања података из .*csv* фајла унутар апликације. (Напомена: .*csv* формат је обавезан. Остали формати, попут *excel-a* су опциони.)
* Визуелизација учитаних података – њихов приказ у табеларном облику.
* Приказ основних статистичких показатеља. Рецимо, за сваку колону је добро приказати њен минимум, максимум, медијану, средњу вредност...
* Приказ корелационе матрице (која колона је у корелацији са којом).
* Корисник треба да има могућност да изабере улазе у мрежу.
* Категоријске величине у табели је потребно енкодирати. Корисник има могућност избора методе енкодирања.
* Дефинисање архитектуре мреже (хиперпараметри): колико слојева неурона ће бити, која активациона функција у неурону ће се користити, који алгоритам за обуку ће се користити. Све ово представља подешавање параметара пре обучавања.
* Покретање прорачуна (алгоритма за обучавање).
* Током обучавања мора постојати визуелизација прогресије. Корисник може да види како се грешка мреже током обучавања смањује.
* Након завршетка обуке, врши се тестирање мреже. (Напомена: наш је избор да ли ћемо приликом увожења података податке одмах поделити на скуп за обуку и скуп за тестирање или ћемо након обуке додати скуп за тестирање.)
* Визуелизација тога колико су добри резултати мреже (разлика онога што дају подаци и што даје мрежа).
* Врши се мерење грешке, при чему корисник може да изабере да ли је у питању *MSE, RMSE*…

# Пожељне функционалности

* Корисник има могућност да запамти шта је до сада урадио, па да следећи пут када уђе у апликацију настави са решавањем проблема/експеримента.
* Логовање корисника у апликацији. Сваки улоговани корисник може да види само своје сачуване експерименте на којима је радио.
* Корисник може да изабере који проценат података ће ући у скуп за учење, а који проценат у скуп за валидацију. (Напомена: добро је да подаци буду измешани)

# Опционе функционалности

* Креирање више модела машинског учења за исти експеримент и поређење модела (која мрежа је дала бољи резултат, тј. мању грешку). Модели се могу разликовати у броју слојева мреже, врсти активационих функција. Треба визуелно приказати разлике међу резултатима модела.
* Корекција података. Уколико корисник примети да неки податак није коректно унет (рецимо за висину је стављена вредност 3,8), има могућност да директно у апликацији измени вредност тог податка, без измене и поновног учитавања .*csv* фајла. Такође, уколико вредност неког податка није унета, корисник је може унети сам.
* Апликација треба првенствено да решава регресионе проблеме, али је велики плус уколико може да решава и класификационе проблеме.