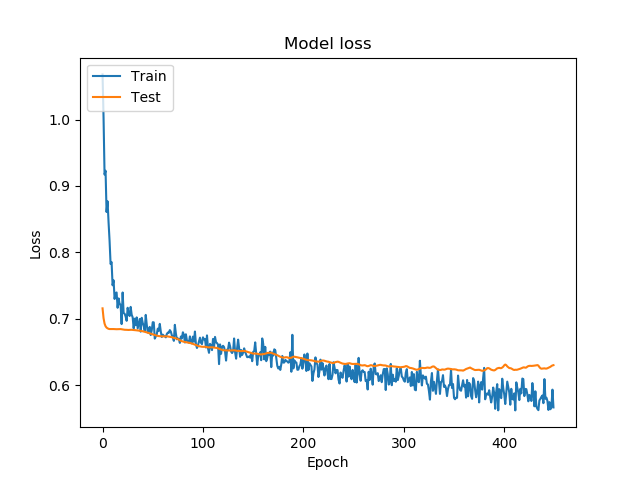
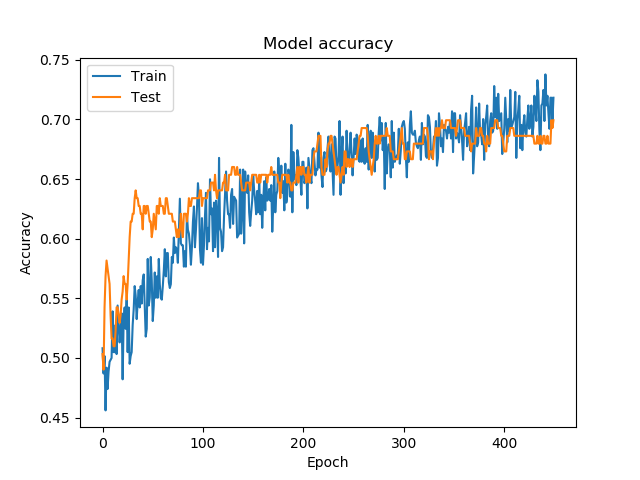
Общий алгоритм работы программы(модели)

Программа построчно (для целостности данных) читает переделанный из *.xlxs* в *.csv* файл, проверяя присутствуют ли *nan, null* или *inf* в данной строке. После чего сохраняет запись в *Record()* (класс записей) и помещает эту запись в список. После чего этот список перебирается и на его основе создаётся словарь, ключами которого служат *subs\_id,* а значениями списки классов *Record().* Такая архитектура позволяет по *subs\_id* обратиться к данным любого пользователя (находящегося в словаре). После этого программа проходится по всем *subs\_id* и их значениям – спискам, в каждом элементе списка программа берёт поле *tariff\_id* и составляет из них *set()* – набор уникальных тарифов, и если он больше 1, то этот пользователь менял свой тариф… После того, как стало известно, какие абоненты меняли тариф, их *subs\_id* помещаются в отдельный список - *positive\_id*, остальные же – в *non\_positive\_id*. *~~После того, как мы нашли тех кто менял тариф, мы решили посмотреть в чём же принципиальные отличия между ними, и теми кто тариф не менял, посмотреть графики можно в папках «positive\_plt» , «negative\_plt» или же единой картинкой Trends-Motive.jpg~~* Но для того, чтобы модель было легче обучать и она работала адекватно, соотношение позитивных примеров и отрицательных, должно быть 1:1. Поэтому был разработан алгоритм, который берёт ключи из списка *non\_positive\_id* без повторений и записывает их в список *negative\_id*. После, программа совмещает *negative\_id* и *positive\_id*, в единый словарь, ставя каждому *subs\_id* соответствие списка из которого он был взят 0 - *negative\_id*, 1 - *positive\_id* и перемешивает их (для того и нужны были ключи, что бы была возможность понимать к какой группе относился абонент), после данные каждого пользователя обрабатываются по принципу: «если на следующую дату значение этого поля (класса *Record()*) больше чем на текущую, то тенденция восходящая => 1, иначе => 0». **В конечном счёте получится список состоящий из 0 и 1, который и нужно подавать на вход нейросети(модели) в формате *np.array().*** Далее эти списки из таких списков разделяются на тренировочную и тестовую выборки, после чего нейросеть обучается… Т.к. negative\_id каждый раз разный, то и результаты при обучении отличаются, худший - 60.142, средний 64.3515, лучший 69.1(был единственный раз и модель не сохранилась), к сожалению более лучших результатов у нас добиться не получилось. Для более лучших результатов нужно более чётко отделить нейтральных абонентов(которых полностью устраивает их текущий тариф), от тех кто хочет сменить поставщика услуг.

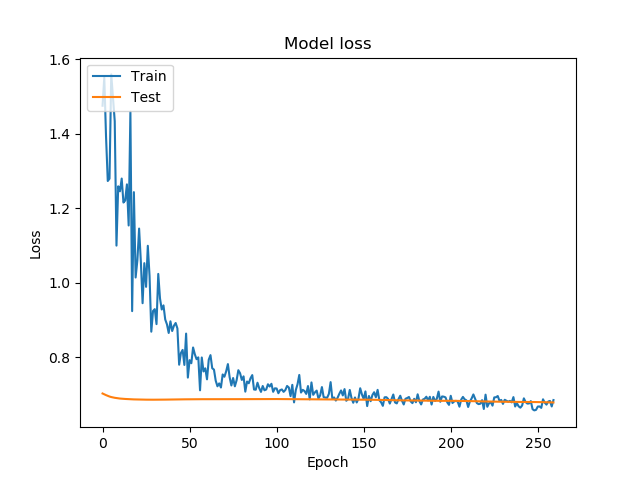
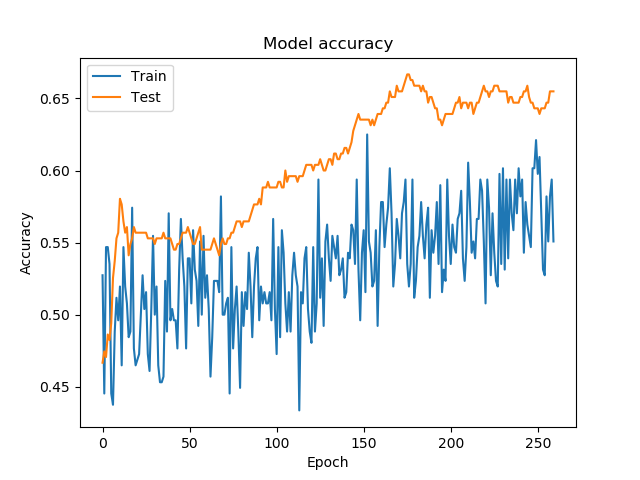
Вот несколько логов обучения и тестирования модели

Основным файлом в проекте является model.py, для тестирования надо перетащить выбранную вами модель .h5 в проект закомментировать 24-56 и раскоментировать 59-62

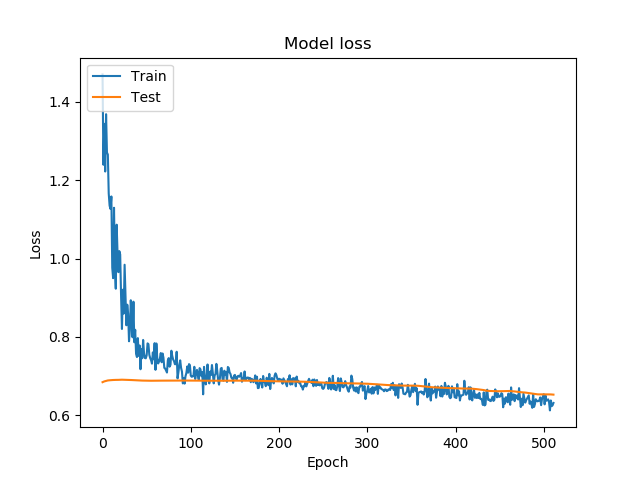
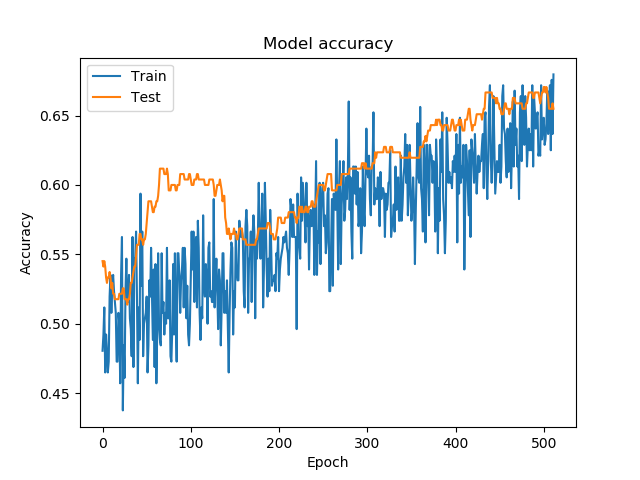
№1



№2



№3



В нашу модель могут быть загружены ВСЕ данные но часть из них закомментирована(попробуйте раскоментировать, обучить посмотрите как изменятся предсказания), т.к. основную роль играют support\_4G, calls\_in\_CNT, calls\_out\_CNT, duration\_out\_min, data\_traffic\_MB, а это основные потребности современного человека, ему важно совершать звонки, пользоваться интернетом и ему очень бы хотелось, чтобы его оператор поддерживал 4G. Поэтому ООО «Мотив» развивает у себя эти направления.