Многослойный персептрон

Введение в глубокое обучение

Цель проекта

Цель этого проекта – осуществить первый подход к искусственным нейронным сетям и реализовать алгоритмы, лежащие в основе их обучения. В то же время вам придется вспомнить вычисление производной сложной функции и основы линейной алгебры, поскольку они являются незаменимыми математическими инструментами для успешной реализации проекта.

Основные инструкции

Язык программирования – Python. Вы можете организовывать и именовать свои файлы по своему усмотрению, соблюдая при этом ограничения, перечисленные ниже.

Не допускается использование библиотек, реализующих искусственные нейронные сети, или базовые алгоритмы (например, Keras), все необходимо кодировать с нуля. Однако можно использовать библиотеки линейной алгебры и графические библиотеки для отображения кривых обучения.

Немаловажная часть оценки будет основана на вашем понимании фазы обучения и базовых алгоритмов, лежащих в ее основе. Вы должны уметь объяснять понятия прямого распространения, обратного распространения и градиентного спуска. Баллы будут начисляться в зависимости от ясности ваших объяснений. Кроме этого вы должны уметь рассказать об архитектуре нейронной сети, ее составных компонентах, а также об особенностях, связанных с решением задачи классификации.

Данные

Набор данных представлен в файле data.csv. Это файл CSV из 32 столбцов, столбец «диагноз» – это метка класса, которую вы хотите предсказать, учитывая все остальные характеристики экземпляра, это может быть либо значение M (malignant), либо B (benign) (злокачественное или доброкачественное).

Параметры набора данных описывают характеристики ядра клетки молочной железы, извлеченной с помощью тонкоигольной биопсии (для более подробной информации перейдите сюда).

Как вы уже знаете, важной составляющей в работе с машинным обучением, является предварительный анализ данных. Поэтому прежде чем приступить к реализации алгоритма обучения нейронной сети и решению задачи, познакомьтесь с вашими данными, проведите статистический и визуальный анализ, сделайте выводы.

Кроме этого вам нужно самостоятельно разделить свой набор данных на две части: одну для обучения и одну для проверки. Помните, что данные являются необработанными и должны быть предварительно обработаны перед использованием на этапе обучения.

Основная задача

Необходимо реализовать многослойный персептрон и обучить вашу модель для задачи классификации.

Ваша реализация нейронной сети должна содержать по крайней мере два скрытых слоя по умолчанию.

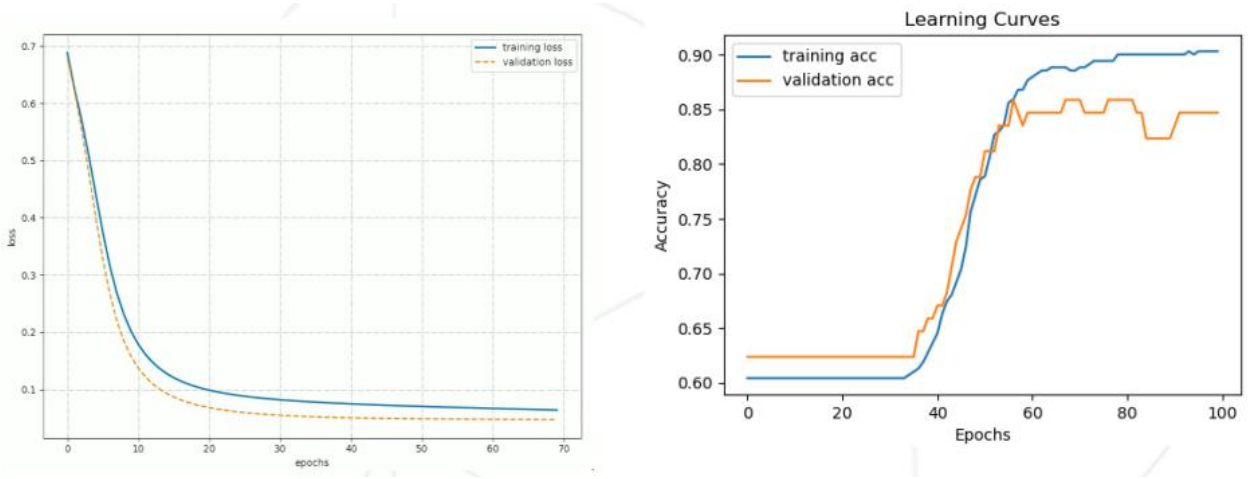
Идея состоит в том, чтобы сделать вашу программу немного более модульной (то есть должна быть возможность конфигурировать ваш многослойный персептрон по собственному усмотрению), например, с помощью файла или аргументов программы, например,

```
cat example_network_modularity.txt

network = model.createNetwork([
    layers.DenseLayer(input_shape, activation='sigmoid'),
    layers.DenseLayer(24, activation='sigmoid', weights_initializer='heUniform'),
    layers.DenseLayer(24, activation='sigmoid', weights_initializer='heUniform'),
    layers.DenseLayer(24, activation='sigmoid', weights_initializer='heUniform'),
    layers.DenseLayer(output_shape, activation='softmax', weights_initializer='heUniform')
])

model.fit(network, data_train, data_valid, loss='binaryCrossentropy', learning_rate=0.0314, batch_size=8,
    epochs=84)
```

```
python train.py --layer 24 24 24 --epochs 84 --loss binaryCrossentropy --batch_size 8 --learning_rate
    0.0314
```

Вам также необходимо реализовать функцию softmax на выходном слое, чтобы получить выход в виде вероятностного распределения.

Необходимо будет также показать два графика кривой обучения, отображаемых в конце фазы обучения (вы можете использовать любую библиотеку для этой цели), например,



Вы должны разработать три программы:

- Программа для разделения набора данных на две части, одну для обучения и другую для проверки
- Программа обучения
- Программа прогнозирования

(или вы можете разработать одну программу с возможностью вызова каждой отдельной части).

Чтобы визуализировать производительность вашей модели во время обучения, необходимо отображать на каждой эпохе метрики обучения и проверки, например,

```
python mlp.py --dataset data_training.csv
x_train shape : (342, 30)
x_valid shape : (85, 30)
epoch 01/70 - loss: 0.6882 - val_loss: 0.6788
...
epoch 39/70 - loss: 0.0750 - val_loss: 0.0406
epoch 40/70 - loss: 0.0749 - val_loss: 0.0404
epoch 41/70 - loss: 0.0747 - val_loss: 0.0400
...
epoch 70/70 - loss: 0.0640 - val_loss: 0.0474
> saving model './saved_model.npy' to disk...
```

Для программы разделения данных используйте один и тот же seed для получения повторяемого результата, поскольку в игру вступают множество случайных факторов (например, инициализация весов и смещения)

- Обучающая программа должна использовать алгоритм обратного распространения ошибки и градиентный спуск для обучения на обучающем наборе данных и сохраняет модель (архитектуру сети и веса) в конце ее выполнения.
- Программа прогнозирования загружает веса, полученные на предыдущем этапе, выполняет прогноз на заданном наборе (который также будет загружен), и затем оценивает его с помощью бинарной кросс-энтропии, см., например, здесь.

Сдача проекта

Рабочие программы с исходным кодом должны быть размещены в вашей индивидуальной папке на Яндекс Диске (под студенческой учетной записью) Основы Python_Фамилия\Многослойный персептрон. В этой же папке необходимо разместить отчет, оформленный средствами LaTex или Word с описанием вашей работы (необходимо включить формулы, код, выкладки и рассуждения, которыми вы пользовались при реализации проекта).

Необходимо предоставить доступ к указанной папке Основы Python_Фамилия преподавателю.

Время на выполнение – 3 недели. Последний срок размещения проекта на Яндекс Диске **31.12.2024**. Проверка проекта осуществляется очно на занятии после новогодних праздников.