JULIA, ИЛИ ЭФФЕКТИВНОЕ ПРОГРАММИРОВАНИЕ ОБЩЕГО НАЗНАЧЕНИЯ, ДЛЯ РЕШЕНИЯ ЗАДАЧ МАШИННОГО ОБУЧЕНИЯ И СТАТИСТИЧЕСКИХ ДАННЫХ

А.С. Монасова, О.И. Захарова

Поволжский государственный университет телекоммуникаций и информатики, Самара, Россия

Обоснование. Популярность Python несравненно растет по сей день. Впрочем, если услышать мнение экспертов и людей, использующих данный язык программирования ежедневно, то преимущественно будут указаны недостатки, нежели преимущества. Данные минусы начинаются с количества времени на выполнение какой-либо задачи и заканчиваются чрезмерным тестированием. Все это подталкивает опытных программистов осваивать новые языки, одним из которых является Julia [1].

Цель — изучить эффективное программирование общего назначения.

Методы. Создание нового языка программирования предполагает новые возможности. Каждый изобретатель планирует избавиться от недостатков предыдущих видов, но сохранить их достоинства. Таким же принципом руководствовались создатели Julia. Вместо замены какого-то конкретного языка, они хотели превзойти всех предыдущих. Говоря о Julia невозможно не сказать про достоинства данного языка [2]. Они считаются его визитной карточкой и тем самым он во многом превосходит многие предыдущие виды. Данными плюсами являются:

- скорость (этот язык разрабатывался для достаточно высокой производительности);
- общность (язык программирования облегчает выражение многих функциональных шаблонов);
- технологичность (Julia превосходна в вычислениях и прекрасно подходит для математики);
- динамичность (язык программирования поддерживает достаточно динамичную типизацию и ведет себя как язык сценариев).

Результаты. Одно из преимуществ данного языка программирования можно увидеть на примере вычисления первых 10 000 простых чисел. Сравнение будет произведено между Python и Julia. На двух взятых кодах будет протестирована скорость выполнения выше прописанной задачи (см. таблицу).

Таблица. Код программирования Python и Julia

Python	Julia

def n_primes(n:int)->list:
  primes = []
  i = 2
  while len(primes) < n:
  prime_bool = True
  for j in range(2,i//2+1):
  if i%j == 0:
  prime_bool = False
  if prime_bool == True:
  primes.append(i)
  i += 1
  return primes
	function n_primes(n::Int64)
	  primes = Int64[]
	  i::Int64 = 2
	  while size(primes)[1] < n
	  prime_bool::Bool = true
	  for j = 2:i÷2
	  if i%j == 0
	  prime_bool = false
	  end
	.....
	end
	>> @time n_primes(10000)

Тестирование этих двух кодов показало, что, выполняя аналогичную функцию, Python делает это в разы медленнее, а конкретнее 2 мин 47 с, в то время как Julia это делает за 8 с. В настоящее время Julia широко применяется в машинном обучении. В основном она фокусируется на таких областях как: компьютерное зрение, графическая аналитика, обработка естественного языка и сигналов [3]. Язык программирования повсеместно применяется в статистических вычислениях, анализе данных и многом другом. Открытый

код дает ему массу возможностей, которые он прекрасно использует. Важно отметить, что Julia имеет все возможные библиотеки машинного обучения для решения всевозможных задач, примерами которых могут быть:

- Mocha.jl (пакет глубокого обучения, написанный полностью на Julia. Он может взаимодействовать с основными функциональными возможностями без необходимости включать внешние зависимости);
- SciKitLearn.jl (оболочка написана на Julia для библиотеки Python);
- указанные библиотеки являются лишь примером из большого списка библиотек машинного обучения, используемых Julia.

Выводы. Несомненно, Julia — это молодой, но довольно перспективный язык программирования. Нельзя сказать, что это аналог всеми любимого Python [4]. Тем не менее можно отметить, что на мировом рынке появляется стоящий конкурент. Julia имеет свои недостатки, как и абсолютно любой язык программирования, но масштаб преимуществ заставляет задуматься о будущем решения задач машинного обучения и статистических данных.

Ключевые слова: Julia; Python; код программирования; машинное обучение; статистические вычисления; анализ данных.

Список литературы

1. Bezanson J., Edelman A., Karpinski S., Shah V.B. Julia: A Fresh Approach to Numerical Computing // SIAM review. 2017. Vol. 59, No. 1. DOI: 10.1137/141000671
2. Lauwens B., Downey A.B. Think Julia: how to think like a computer scientist. O'Reilly Media Inc., 2019. 450 p.
3. geeksforgeeks.org [Электронный ресурс]. Introduction to Machine Learning in Julia // GeeksforGeeks. Доступ по ссылке: https://www.geeksforgeeks.org/introduction-to-machine-learning-in-julia/
4. Bezanson J., Karpinski S., Shah V.B., Edelman A. Julia: A Fast Dynamic Language for Technical Computing // arXiv. 2012. Vol. 1. DOI: 10.48550/arXiv.1209.5145

Сведения об авторах:

Анна Сергеевна Монасова — студентка, группа ИСТ-91, факультет информационных систем и технологий; Поволжский государственный университет телекоммуникаций и информатики, Самара, Россия. E-mail: monasovas83@gmail.com

Оксана Игоревна Захарова — научный руководитель, кандидат технических наук, доцент; доцент кафедры информационных систем и технологий; Поволжский государственный университет телекоммуникаций и информатики, Самара, Россия. E-mail: xeniya-luna@list.ru