1. Развертывание среды разработки Julia и решение одной задачи

Цель работы: изучить процесс развертывания среды программирования Julia в Visual Studio Code и выполнить задачу прогнозирования цен на жильё с использованием линейной регрессии.

1. Развертывание среды разработки

Этап 1. Установка Julia
Julia была скачана с официального сайта и установлена в каталог без кириллических символов:

C:\Users\Default\Julia-1.11.7

Это важно, так как расширение Julia в VS Code может некорректно работать, если путь к Julia содержит русские буквы.

Этап 2. Настройка Visual Studio Code
В VS Code было установлено расширение Julia из Marketplace.

Этап 3. Указание пути к Julia в настройках VS Code
В файле settings.json было добавлено:

"julia.executablePath": "C:\\Users\\Default\\Julia-1.11.7\\bin\\julia.exe",

"julia.symbolCacheDownload": true,

"julia.enableIJulia": true

Этап 4. Проверка работы

- Открыл терминал PowerShell в VS Code.
- Проверил командой julia --version.
- Julia успешно запустилась, что подтвердило корректность установки и настройки среды.

Замечание: первоначально Julia не запускалась в терминале VS Code, хотя работала в отдельном PowerShell. Проблема заключалась в том, что VS Code был открыт до добавления Julia в PATH, поэтому старые окна VS Code не видели новый путь. Решение: полностью закрыть и перезапустить VS Code.

2. Решение задачи: прогнозирование цен на недвижимость

Этап 1. Установка необходимых пакетов

```
import Pkg
Pkg.add("CSV")
Pkg.add("DataFrames")
Pkg.add("GLM")
Pkg.add("Statistics")
```

Этап 2. Подключение пакетов

```
julia> Pkg.add("CSV")          # для работы с CSV
   Resolving package versions...
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Project.toml`
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Manifest.toml`

julia> Pkg.add("DataFrames")   # для работы с таблицами
   Resolving package versions...
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Project.toml`
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Manifest.toml`

julia> Pkg.add("GLM")          # для линейной регрессии
   Resolving package versions...
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Project.toml`
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Manifest.toml`

julia> Pkg.add("Statistics")   # для метрик, например, MSE
   Resolving package versions...
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Project.toml`
  No Changes to `C:\Users\Дмитрий\.julia\environments\v1.11\Manifest.toml`
```

Этап 3. Загрузка данных

```
julia> url = "https://raw.githubusercontent.com/selva86/datasets/master/BostonHousing.csv"
"https://raw.githubusercontent.com/selva86/datasets/master/BostonHousing.csv"

julia> housing = CSV.File(download(url)) |> DataFrame
506×14 DataFrame
```

Row	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
	Float64	Float64	Float64	Int64	Float64	Float64	Float64	Float64	Int64	Int64	Float64	Float64	Float64	Float64
1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24.0
2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
3	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
6	0.02985	0.0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
7	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
501	0.22438	0.0	9.69	0	0.585	6.027	79.7	2.4982	6	391	19.2	396.9	14.33	16.8
502	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.4

Этап 4. Разделение данных на обучающую и тестовую выборки

```
Random.seed!(1234)
n = nrow(housing)
shuffle_idx = shuffle(1:n)
train_idx = shuffle_idx[1:round(Int, 0.8*n)]
test_idx = shuffle_idx[round(Int, 0.8*n)+1:end]
train_data = housing[train_idx, :]
test_data  = housing[test_idx, :]
```

```
julia> test_idx = shuffle_idx[round(Int, 0.8*n)+1:end]
101-element Vector{Int64}:
 198
  23
 490
 239
 224
```

```
julia> train_data = housing[train_idx, :]
405×14 DataFrame
```

Row	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
	Float64	Float64	Float64	Int64	Float64	Float64	Float64	Float64	Int64	Int64	Float64	Float64	Float64	Float64
1	0.06047	0.0	2.46	0	0.488	6.153	68.8	3.2797	3	193	17.8	387.11	13.15	29.6
2	0.03961	0.0	5.19	0	0.515	6.037	34.5	5.9853	5	224	20.2	396.9	8.01	21.1
3	0.08873	21.0	5.64	0	0.439	5.963	45.7	6.8147	4	243	16.8	395.56	13.45	19.7
4	0.09164	0.0	10.81	0	0.413	6.065	7.8	5.2873	4	305	19.2	390.91	5.52	22.8
5	5.58107	0.0	18.1	0	0.713	6.436	87.9	2.3158	24	666	20.2	100.19	16.22	14.3
6	0.15876	0.0	10.81	0	0.413	5.961	17.5	5.2873	4	305	19.2	376.94	9.88	21.7
7	22.5971	0.0	18.1	0	0.7	5.0	89.5	1.5184	24	666	20.2	396.9	31.99	7.4

Этап 5. Построение модели линейной регрессии

```
julia> formula = @formula(medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax + ptratio + b + lstat)
FormulaTerm
Response:
  medv(unknown)
Predictors:
  crim(unknown)
  zn(unknown)
  indus(unknown)
  chas(unknown)
  nox(unknown)
  rm(unknown)
  age(unknown)
  dis(unknown)
  rad(unknown)
  tax(unknown)
  ptratio(unknown)
  b(unknown)
  lstat(unknown)
```

```
julia> lm_model = lm(formula, train_data)
StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Vector{Float64}}, GLM.DensePredChol{Float64, LinearAlgebra.CholeskyPivoted{Float64, Matrix{Float64}, Vector{Int64}}}}, Matrix{Float64}}

medv ~ 1 + crim + zn + indus + chas + nox + rm + age + dis + rad + tax + ptratio + b + lstat

Coefficients:
```

	Coef.	Std. Error	t	Pr(>|t|)	Lower 95%	Upper 95%
(Intercept)	38.7884	5.60808	6.92	<1e-10	27.7627	49.8142
crim	-0.111497	0.0337156	-3.31	0.0010	-0.177783	-0.04521
zn	0.0492532	0.0159303	3.09	0.0021	0.0179334	0.0805729
indus	0.0467366	0.0710104	0.66	0.5108	-0.0928734	0.186347
chas	1.70812	1.01993	1.67	0.0948	-0.297116	3.71336
nox	-19.1918	4.22822	-4.54	<1e-05	-27.5047	-10.8789
rm	3.72133	0.455476	8.17	<1e-14	2.82584	4.61682
age	-0.00380972	0.0149752	-0.25	0.7993	-0.0332518	0.0256324
dis	-1.54018	0.222611	-6.92	<1e-10	-1.97785	-1.10252
rad	0.307443	0.0747036	4.12	<1e-04	0.160572	0.454314
tax	-0.0134704	0.0043543	-3.09	0.0021	-0.0220312	-0.00490961
ptratio	-0.965717	0.149674	-6.45	<1e-09	-1.25998	-0.67145
b	0.00906793	0.00289971	3.13	0.0019	0.00336695	0.0147689
lstat	-0.51588	0.0564886	-9.13	<1e-17	-0.62694	-0.404821

3. Результаты работы

- Модель линейной регрессии была успешно построена.
- Рассчитаны предсказанные значения цен на жильё.
- Вычислена среднеквадратичная ошибка (MSE) на тестовой выборке, что позволяет оценить точность модели.

4. Вывод

Среда разработки Julia была успешно развернута в Visual Studio Code, включая корректное указание пути к исполняемому файлу и настройку терминала. Задача прогнозирования цен на жильё решена с использованием линейной регрессии, что демонстрирует работоспособность установленной среды и корректность построенной модели.