Министерство науки и высшего образования РФ

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

«Санкт-Петербургский государственный электротехнический

университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»

(СПбГЭТУ «ЛЭТИ»)

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

Индивидуальное Домашнее Задание

По курсу «Оптимизация и многокритериальный выбор»

на тему «Статистика на Python»

|  |  |  |
| --- | --- | --- |
| Студент гр. 0308 |  | Радабольский В.С. |

Санкт-Петербург

2022

**Содержание**

[**Цель работы** 3](#_Toc1)

[**Задание** 3](#_Toc2)

[**Стриптиз** 3](#_Toc3)

[**Стек технологий** 4](#_Toc4)

[**Блок-схема** 4](#_Toc5)

[**Распределение обязанностей** 10](#_Toc6)

[**Построение регрессионной модели** 10](#_Toc7)

[**Дополнительно** 15](#_Toc8)

[**Выводы** 17](#_Toc9)

[**Список литературы** 18](#_Toc10)

[**Приложение А** 18](#_Toc11)

[**Приложение B** 20](#_Toc12)

# **Цель работы**

Реализовать работу, описанную в ВКР Тумановой В.Д. по теме “Статистическая модель производительности ноутбуков”, на языке Python с использованием библиотек с открытым исходным кодом.

# **Задание**

Необходимо выбрать стек технологий для реализации системы, построить блок-схемы алгоритма работы системы, распределить выполнение работы между участниками бригады, построить статистическую модель, описанную в ВКР Тумановой В.Д.

# **Стриптиз**

До выполнения работы:

* Прочитана ВКР Тумановой В.Д.
* Выполнен поиск библиотек с открытым исходным кодом для реализации работы Тумановой В.Д.
* Выполнен поиск и ознакомление с описаниями алгоритма шагового отбора переменных

Выполнение работы:

* Общий алгоритм выполнения работы представлен в виде блок-схем
* Распределение обязанностей между участниками бригады
* Создан репозиторий github для совместной работы с участниками бригады, где были представлены блок-схемы, их краткое описание, а также указаны ссылки на библиотеки, которые могут помочь при реализации, и их официальную документацию
* Реализовано построение модели методом шагового отбора, описанным в ВКР Тумановой В.Д.
* Выполнено тестирование полученной модели на реальных данных, полученных от бригады А. Пухи

Результат:

* Построена система с автоматической предобработкой, построением модели, с автоматической проверкой предположений о модели.
* Система протестирована на реальных данных

# **Стек технологий**

Язык программирования: Python 3

Платформа: Jupyter Notebook

Система контроля версий: github (<https://github.com/radabolsky/laptop-performance>)

Используемые библиотеки:

|  |  |
| --- | --- |
| **Библиотека** | **Назначение** |
| math | Встроенная в стандартную библиотеку python библиотека для работы с математическими функциями |
| pandas | Библиотека для удобной работы с табличными данными |
| numpy | Библиотека быстрых математических вычислений с возможностью представления данных в виде векторов |
| seaborn | Визуализация табличных данных |
| matplotlib |
| scipy | Библиотека статистических функций |
| statsmodels | Библиотека для построения статистических моделей |

# **Блок-схема**

Общий алгоритм работы выглядит следующим образом:

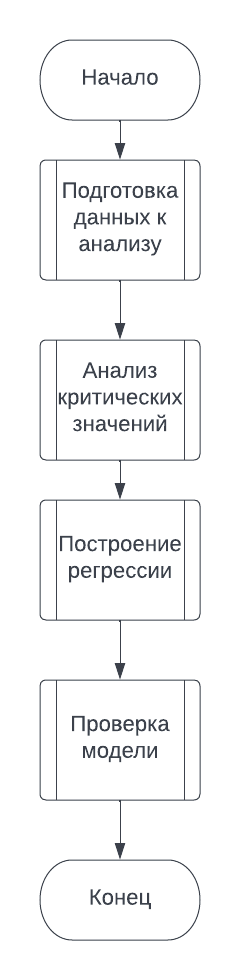


Рисунок . Общий алгоритм

Подготовка данных к анализу:

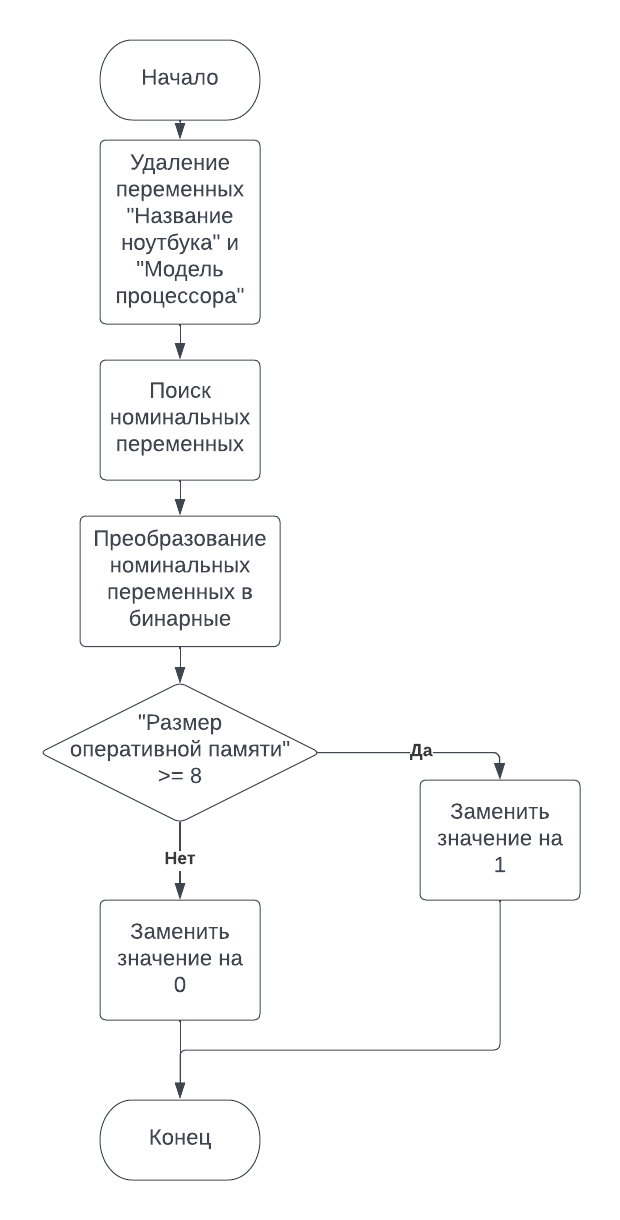


Рисунок . Подготовка данных к анализу

Анализ критических значений:

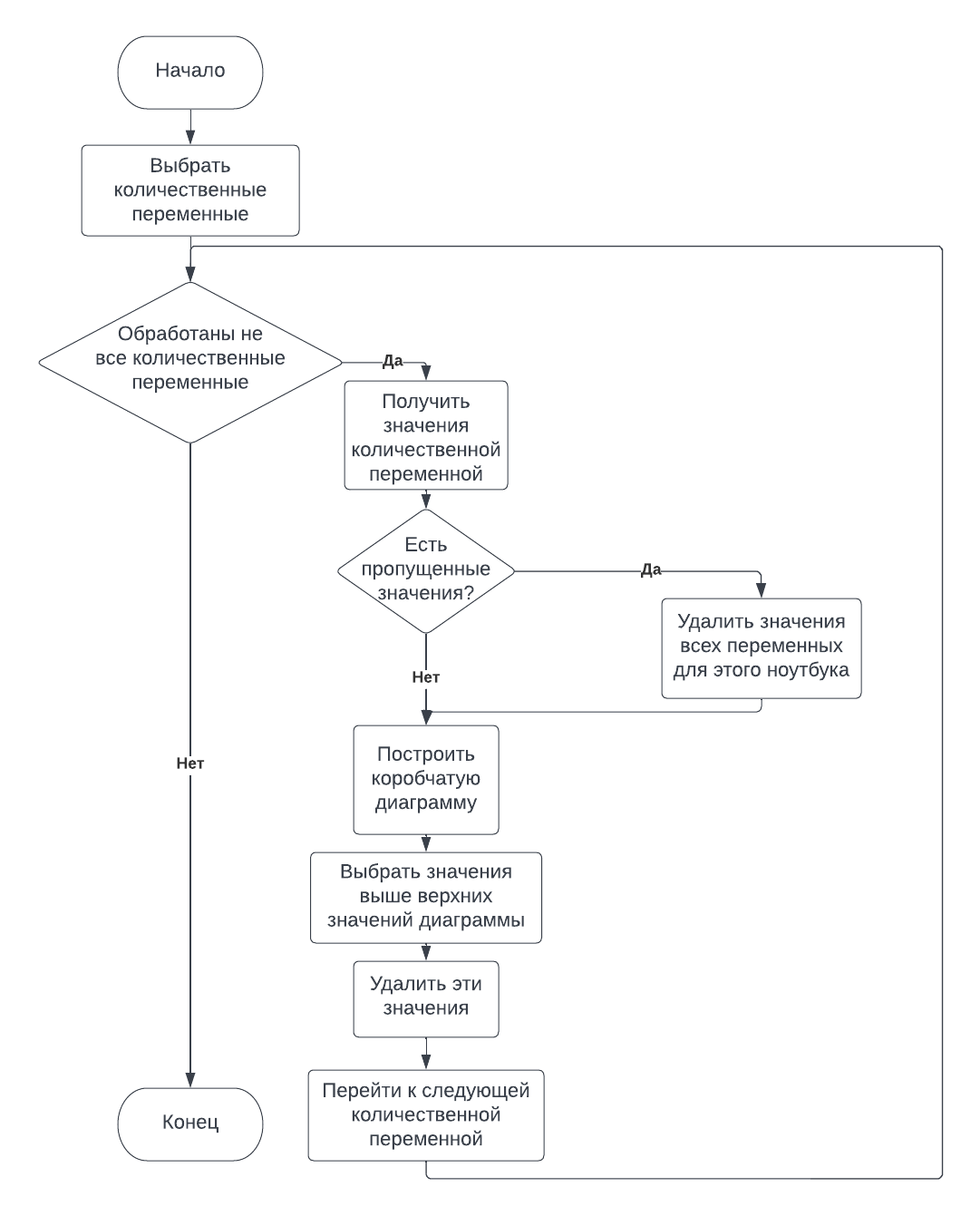


Рисунок . Анализ критических значений

Построение регрессионной модели:

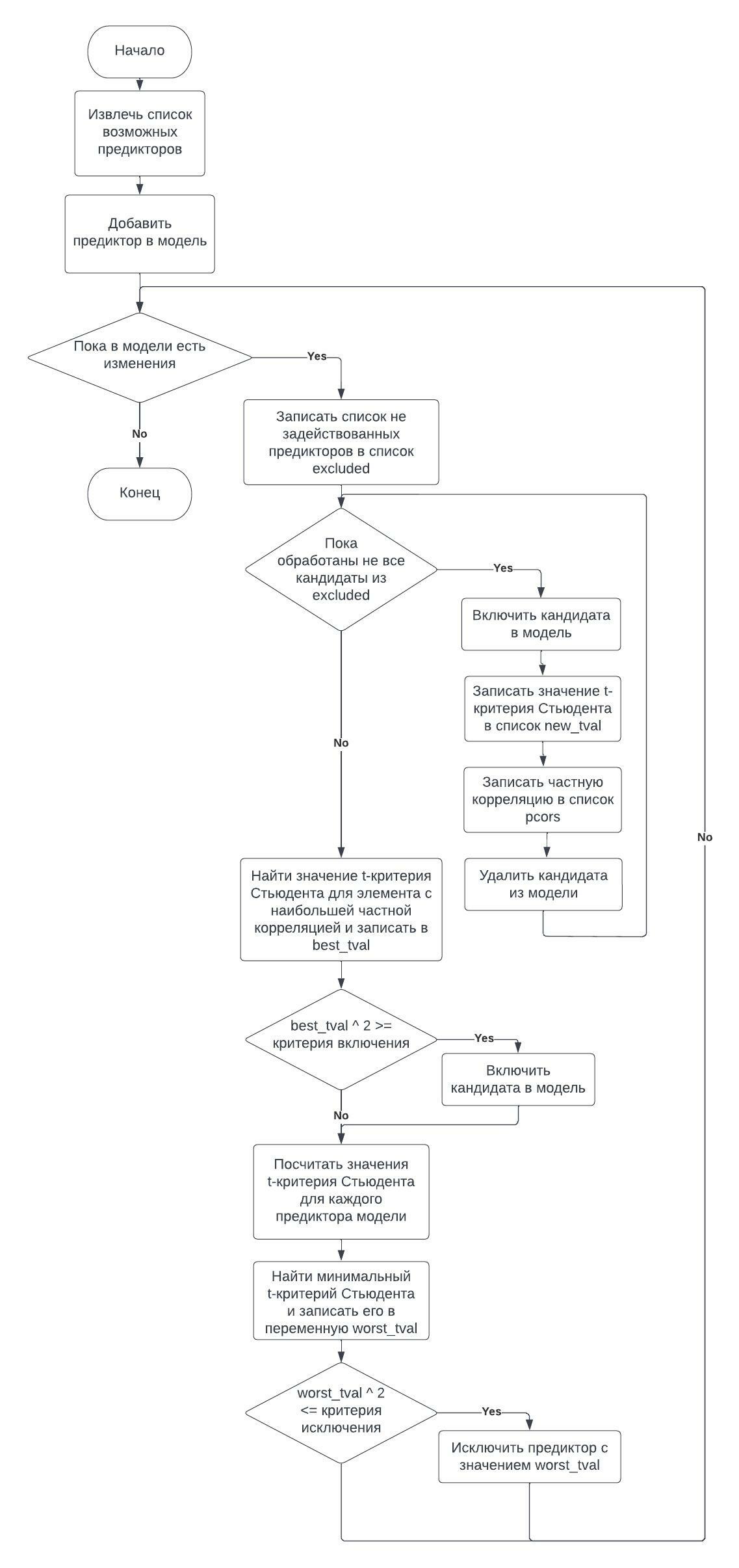


Рисунок . Построение регрессионной модели

Тестирование модели

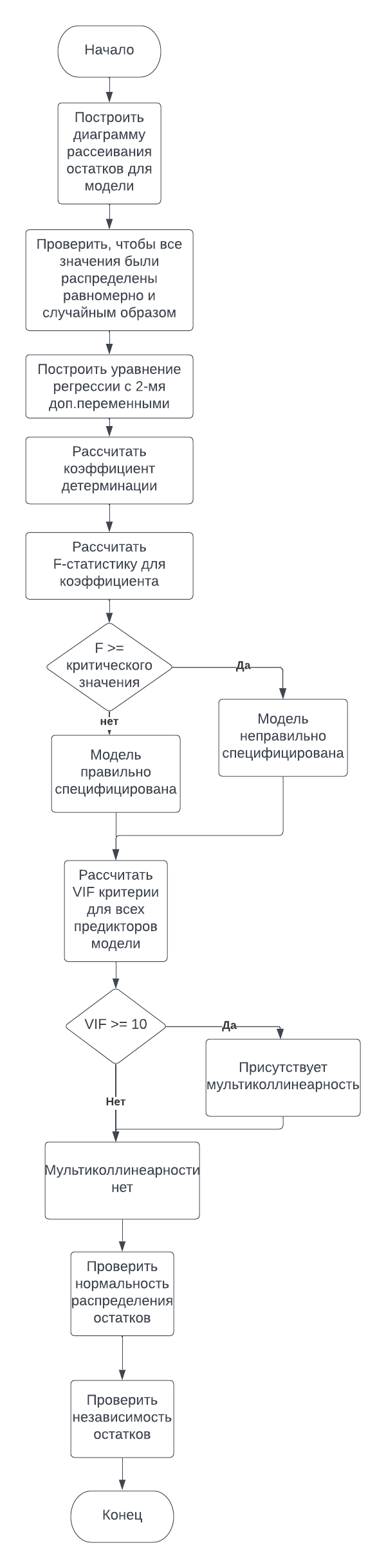


Рисунок . Проверка предположений о модели

# **Распределение обязанностей**

|  |  |
| --- | --- |
| **Шаг алгоритма** | **Ответственный** |
| Подготовка данных к анализу | Лесниченко Александр 0308 |
| Анализ критических значений | Лесниченко Александр 0308 |
| Построение регрессионной модели | Радабольский Владислав 0308 |
| Тестирование модели | Белов Александр 0306 |

# **Построение регрессионной модели**

Для построения модели использовались данные, полученные после предварительной предобработки, во время которой были удалены лишние и шумовые значения, а также все данные преобразованы к числовым (более подробно этот шаг описан в отчете Лесниченко Александра). Фрагмент предобработанных данных изображен на Рисунке 6.



Рисунок . Фрагмент обработанных данных

Данные и их представление полностью совпали с работой Тумановой В.Д.

После предобработки датасет представляет собой 27 ноутбуков с 15 признаками (включая целевую переменную: 'Результаты в бенчмарке PCMark10')

Далее из всех признаков нужно выделить наиболее значимые, которые вносят наибольший вклад в вариацию зависимой переменной. Одним из способов нахождения таких признаков является метод шагового отбора.

Рассмотрим реализацию этого метода в работе Тумановой В.Д. Алгоритм можно представить следующим образом:

*“Добавление предикторов в модель происходит аналогично методу включения: сначала берется предиктор, который имеет наибольшую по модулю корреляцию с зависимой переменной и удовлетворяет критерию ввода, а затем добавляются признаки, удовлетворяющие критерию ввода, с наибольшей частной корреляцией с зависимой переменной. Отличительной чертой данного метода является то, что на каждом шаге исследуются F - критерии каждого предиктора в модели, а не только того предиктора, который был добавлен последним.*

*Таким образом, на каждом шаге, помимо включения новой переменной*

*в уравнение регрессии, также может происходить исключение переменной,*

*которая после добавления других переменных стала вносить вклад в модель*

*меньше заданного уровня. Процедура происходит до тех пор, пока в уравнении не остается переменных, которые можно было бы исключить, а также нет предиктора, не включенного в модель, который бы имел значимую корреляцию с откликом.” [1]*

Критерии включения и исключения были выбраны также согласно ВКР, критерий включения 𝐹 ≥ 3,84 и критерий исключения 𝐹 ≤ 2,71.[1]

В Приложении А содержится полный листинг программы, в котором за реализацию шагового отбора отвечает функция **stepwise\_selection**.

Параметры функции:

* X – данные по всем существующим признакам. Тип: pandas.DataFrame
* y – данные целевой переменной. Тип: pandas.Series
* threshold\_in – критерий включения. Тип: float
* threshold\_out – критерий исключения. Тип: float

Возвращаемое значение:

* included – список предикторов, которые входят в модель. Тип: list
* pivot\_df – сводная таблица с основной информацией о модели на каждом шаге. Тип: pandas.DataFrame

Описание переменных представлено в Таблице 1.

Таблица 1 - Описание переменных функции stepwise\_selection

|  |  |
| --- | --- |
| **Переменная** | **Назначение** |
| included | Список переменных, включенных в модель |
| i | Номер шага алгоритма |
| pivot\_df | Сводная таблица по каждому шагу отбора |
| changed | Индикатор изменения модели. |
| excluded | Признаки, не участвующие в построении модели, из которых выбирается кандидат на добавление |
| new\_tval | Значения t-критерия Стьюдента для каждого кандидата на добавление |
| pcors | Частная корреляция для каждого кандидата |
| new\_column | Кандидат на добавление в модель |
| model | Регрессионная модель |
| best\_tval | Значение t-критерия Стьюдента для кандидата с наибольшей частной корреляцией |
| best\_feature | Название предиктора, удовлетворяющего условию включения в модель |
| model\_info | Основная информация о модели на текущей итерации |
| tvalues | Значения t-критерия Стьюдента для каждого предиктора модели |
| worst\_tval | Предиктор с минимальным значением t-критерия Стьюдента для текущей модели |

Алгоритм:

1. Инициализация предикторов, включенных в модель (included) пустым списком
2. Вход в цикл, который выполняется до тех пор, пока модель не перестает изменяться
   1. Поочередное добавление в текущую модель кандидата из списка еще не включенных в модель предикторов
   2. Запись в список new\_tval значений t-критерия Стьюдента для каждого кандидата
   3. Запись в список pcors модулей частных корреляций каждого кандидата
   4. Выбор кандидата с наибольшим значением модуля частной корреляции и запись этого значения в переменную best\_tval
   5. Сравнение best\_tval с критерием добавления в модель. Если квадрат best\_tval больше или равен критерию добавления (3.84), то включение кандидата в модель
   6. Запись значений t-критерия Стьюдента для каждого предиктора модели в список tvalues
   7. Сравнение минимального значения tvalues с критерием исключения из модели (2.71). Если значение меньше или равно критерия, то исключение признака с этим значением из модели

Для построения модели использовался класс OLS из библиотеки **statsmodels**, а для получения F-критерия - квадрат результата вызова его метода **.tvalues**, возвращающего значения t-статистики. Частная корреляция рассчитывалась с помощью функции **pcorr** из расширения **pingouin** для библиотеки pandas.

Результаты добавления и удаления признаков из модели сохраняются в специальной сводной таблице с помощью функции **get\_pivot** (листинг см. в Приложении А).

В результате множественного регрессионного анализа была получена модель с коэффициентами представленными на Рисунке 7.

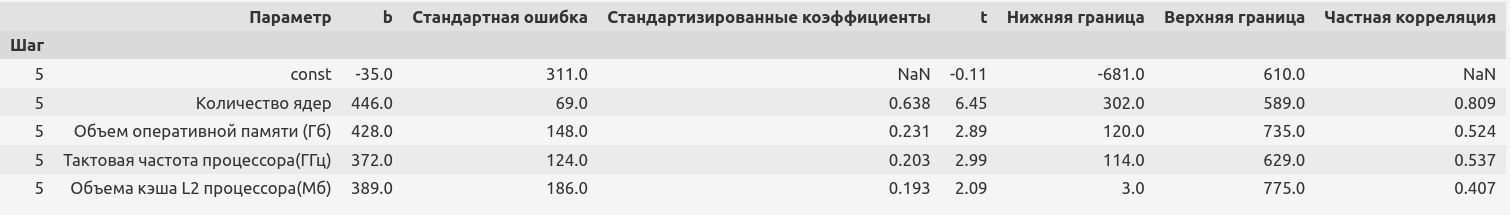


Рисунок 7 - Коэффициенты регрессионной модели

Коэффициенты совпадают с коэффициентами модели из ВКР[1]. Коэффициенты этой модели изображены на Рисунке 8.

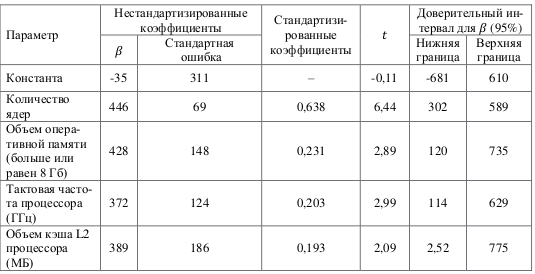


Рисунок 8 - коэффициенты модели из ВКР

На Рисунке 9 приведены данные для коэффициентов моделей на каждом шаге метода шагового отбора.

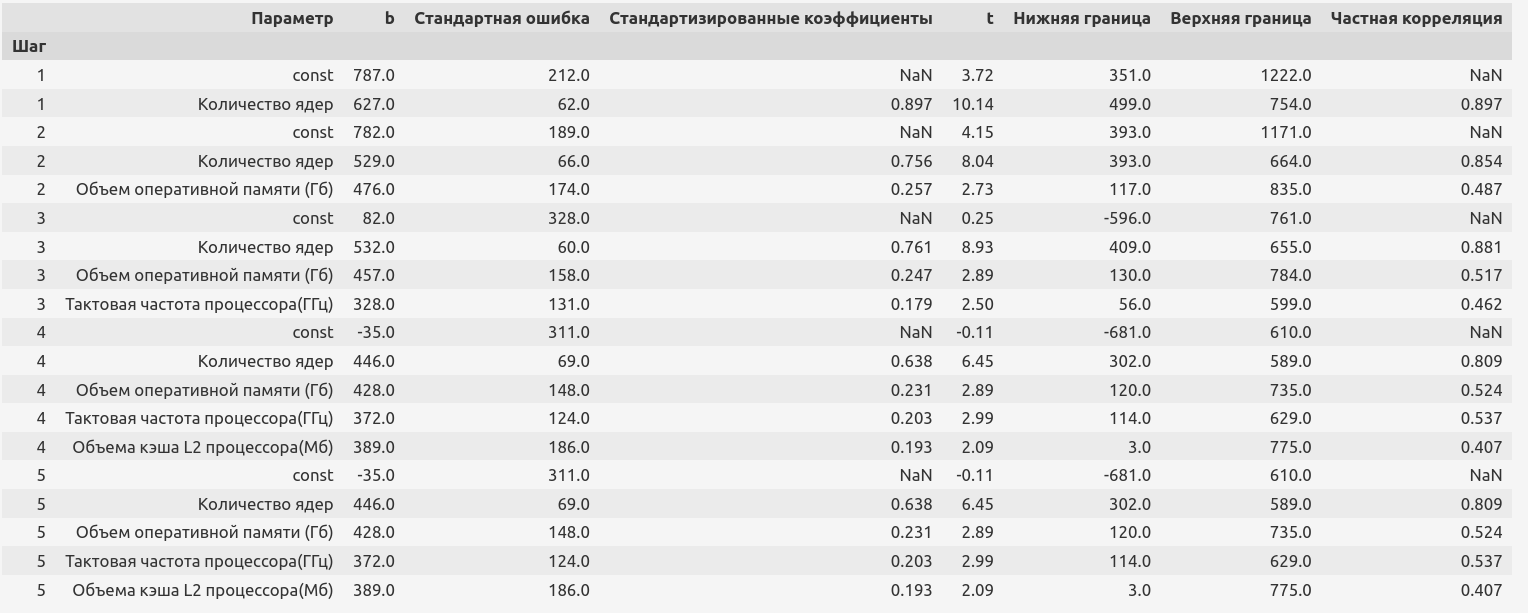


Рисунок 9 - коэффициенты моделей на каждом шаге отбора

Таблица совпадает с результатами, приведенными в ВКР[1]. Коэффициенты моделей из ВКР представлены на Рисунке 10.

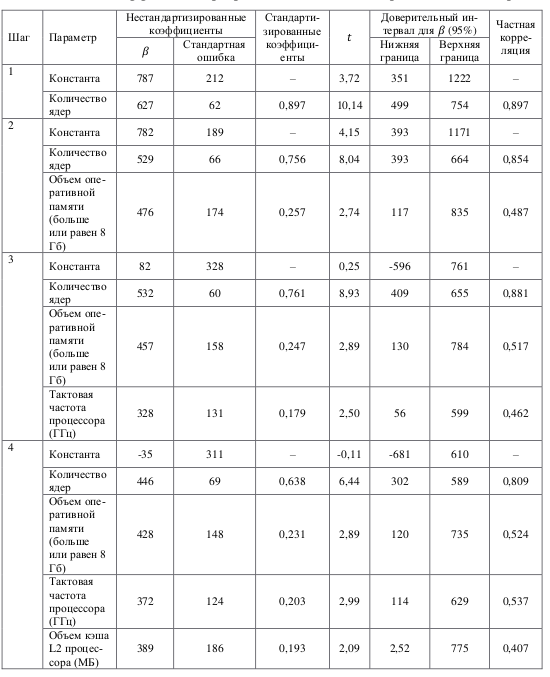


Рисунок 10 - коэффициенты моделей на каждом шаге отбора (ВКР)

# **Дополнительно**

Построенная модель была протестирована на реальных данных полученных от бригады А. Пухи.

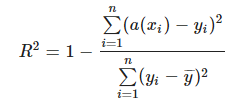
Изначально тестовые данные не подходили по представлению и их требовалось предобработать. Для тестирования модели необходимо было обработать столбцы: количество ядер, объем оперативной памяти, тактовая частота процессора (ГГц), объем кеша L2 процессора (Мб). Данные были обработаны следующим образом:

1. Количество ядер – без изменений
2. Объем кеша L2 процессора – преобразованы из Килобайт в Мегабайты
3. Объем оперативной памяти преобразован согласно алгоритму[1] (значения большие или равные 8 принимаются за 1, остальные - 0)
4. Тактовая частота процессора – без изменений

Предобработанные данные представлены в формате csv в Приложении Б.

В качестве метрик качества использовались такие как:[2]

* Коэффициент детерминации (r2)



* Средняя квадратичная ошибка (MSE)



* Средняя абсолютная ошибка (MAE)



* Средняя абсолютная процентная ошибка



Так как результаты бенчмарков в тестовых данных были разделены на 3 режима работы, метрики качества измерялись для каждого режима в отдельности. Результаты приведены на Рисунке 11.

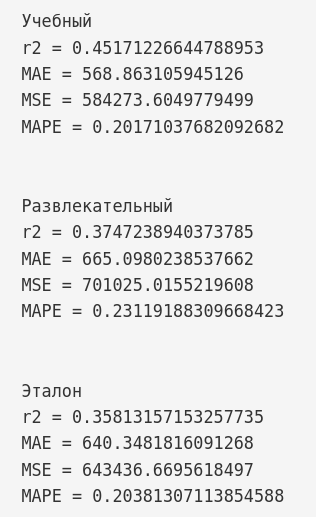


Рисунок 11 - Результаты тестирования модели на реальных данных

# **Выводы**

В ходе выполнения работы была написана система, позволяющая прогнозировать результаты бенчмарков производительности для ноутбуков. Система является копией системы, описанной в ВКР[1], но обладает преимуществами открытого исходного кода. Сравнительный анализ процесса построения модели и модели полученной в результате, показывает, что процесс построения и результат совпадают с предложенным в ВКР[1].

Полученная модель была протестирована на реальных данных. Результаты тестирования показывают, что ошибка на данных, с которыми модель не встречалась во время обучения существенна и требует расширения обучающей выборки для построения более точной модели.

# **Список литературы**

1. ВКР Тумановой В.Д. по теме «Статистическая модель производительности ноутбуков»
2. Оценка качества модели [электронный ресурс] - [Оценка качества в задачах классификации и регрессии — Викиконспекты (ifmo.ru)](https://neerc.ifmo.ru/wiki/index.php?title=%D0%9E%D1%86%D0%B5%D0%BD%D0%BA%D0%B0_%D0%BA%D0%B0%D1%87%D0%B5%D1%81%D1%82%D0%B2%D0%B0_%D0%B2_%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%B0%D1%85_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8_%D0%B8_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D0%B8#.D0.9E.D1.86.D0.B5.D0.BD.D0.BA.D0.B8_.D0.BA.D0.B0.D1.87.D0.B5.D1.81.D1.82.D0.B2.D0.B0_.D1.80.D0.B5.D0.B3.D1.80.D0.B5.D1.81.D1.81.D0.B8.D0.B8)
3. В.В. Стрижов, Е.А. Крымова – «Методы выбора регрессионных моделей»
4. Пример шаговой регрессии [электронный ресурс]- [Шаговая регрессия (пример) (machinelearning.ru)](http://www.machinelearning.ru/wiki/index.php?title=%D0%A8%D0%B0%D0%B3%D0%BE%D0%B2%D0%B0%D1%8F_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D1%8F_%28%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80%29)

# **Приложение А**

Код программы:

*# %%*

from math import sqrt

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

import warnings

warnings.filterwarnings('ignore')

from scipy import stats

import statsmodels.api as sm

import statsmodels.stats.api as sms

from statsmodels.stats.outliers\_influence import OLSInfluence

from statsmodels.stats.outliers\_influence import variance\_inflation\_factor

from scipy.stats import shapiro

from scipy.stats import zscore

from pingouin import partial\_corr

*# %%*

df = pd.read\_csv(r"data/tum\_preprocessing.csv")

*# %%*

label= 'Результа ты в бенчмарке PCMark10'

*# %%*

X\_data = df.drop(label, axis=1)

Y\_data = df.drop(X\_data.columns, axis=1)

*# %%*

def **get\_pivot**(X, model, param, y, num):

pivot\_test = pd.DataFrame()

predictors = model.params.index.tolist()

n = **len**(predictors)

i = np.ones(n).astype(int) \* num

pivot\_test['Шаг'] = pd.Series(i)

pivot\_test['Параметр'] = predictors

pivot\_test['b'] = model.params.values.round()

pivot\_test['Стандартная ошибка'] = model.bse.values.round()

zmodel = sm.OLS(zscore(y), zscore(X[param])).fit()

pivot\_test['Стандартизированные коэффициенты'] = np.concatenate(([np.NAN], zmodel.params.values.round(3)))

pivot\_test['t'] = model.tvalues.values.round(2)

pivot\_test['Нижняя граница'] = model.conf\_int(alpha=0.05)[:][0].values.round()

pivot\_test['Верхняя граница'] = model.conf\_int(alpha=0.05)[:][1].values.round()

corr = pd.concat((X[param],y), axis=1).pcorr().round(3)['Результа ты в бенчмарке PCMark10']

corr.pop('Результа ты в бенчмарке PCMark10')

pivot\_test['Частная корреляция'] = np.concatenate(([np.NAN],corr.values))

return pivot\_test.set\_index('Шаг')

def **stepwise\_selection**(X, y,

threshold\_in=3.84,

threshold\_out = 2.71):

included = list()

i = 1

pivot\_df = pd.DataFrame()

while True:

changed=False

*# forward step*

excluded = list(set(X.columns)-set(included))

new\_tval = pd.Series(index=excluded)

pcors = pd.Series(index=excluded)

for new\_column in excluded:

model = sm.OLS(y, sm.add\_constant(pd.DataFrame(X[included+[new\_column]]))).fit()

new\_tval[new\_column] = model.tvalues[new\_column]

pcors[new\_column] = np.abs(pd.concat((X[included+[new\_column]], y), axis=1).pcorr()['Результа ты в бенчмарке PCMark10'][new\_column])

best\_tval = new\_tval[pcors.argmax()]

if best\_tval \*\* 2 >= threshold\_in:

best\_feature = new\_tval.argmax()

included.append(new\_tval.index[best\_feature])

changed=True

*# backward step*

model = sm.OLS(y, sm.add\_constant(pd.DataFrame(X[included]))).fit()

model\_info = get\_pivot(X, model, included, y, i)

pivot\_df = pd.concat([pivot\_df, model\_info], axis=0)

i += 1

tvalues = model.tvalues.iloc[1:]

worst\_tval = tvalues.min()

if worst\_tval \*\* 2 <= threshold\_out:

changed=True

worst\_feature = tvalues.argmin()

included.remove(worst\_feature)

model = sm.OLS(y, sm.add\_constant(pd.DataFrame(X[included]))).fit()

model\_info = get\_pivot(X, model, included, y, i)

pivot\_df = pd.concat([pivot\_df, model\_info], axis=0)

i += 1

if not changed:

break

return included, pivot\_df

result, res\_df = stepwise\_selection(X\_data, Y\_data)

**print**('resulting features:')

**print**(result)

*# %%*

model = sm.OLS(Y\_data, sm.add\_constant(df[result])).fit()

*# %%*

summary = pd.DataFrame()

summary['R^2'] = model.rsquared

summary['Скорректированный R^2'] = model.rsquared\_adj

summary['F'] = model.fvalue

# **Приложение B**

Количество ядер,Количество логических процессоров (потоков),**Тактовая частота процессора (ГГц),***Максимальная тактовая частота (ГГц),*Объем кэша L2 процессора (Кб),Объем кэша L3 процессора (Кб),Частота оперативной памяти (МГц),**HDD (Гб),SSD(Гб),**Объем видеопамяти (Гб) [дискретной],Объем видеопамяти (Гб) [встроенной],Учебный,**Развлекательный,***Эталон,*Наличие графического ускорителя: дискретный,Тип видеопамяти: DDR4,Тип видеопамяти: GDDR5,**Тип видеопамяти: GDDR6,Тип видеопамяти: LPDDR4X,**Тип видеопамяти: SMA,Оперативная память > 8,benchmark

6,12,**3.0,***4.0,*3072,8192,3200,**0,1,**4.0,0.5,5076,**5082,***5120,*1.0,0,0,**1,0,**0,1,15278

2,4,**2.2,***3.4,*512,4096,2133,**0,1,**0.0,0.128,2523,**2299,***2743,*0.0,0,0,**0,0,**1,0,7565

8,8,**2.0,***4.1,*4096,8192,2667,**0,1,**0.0,8.0,4424,**4371,***4565,*0.0,1,0,**0,0,**0,1,13360

6,12,**2.3,***4.2,*3072,16384,3200,**0,1,**0.0,8.0,5052,**5138,***5204,*0.0,1,0,**0,0,**0,1,15394

4,4,**1.1,***3.3,*1536,4096,3200,**0,1,**0.0,8.0,2176,**2506,***2561,*0.0,1,0,**0,0,**0,1,7243

2,4,**2.5,***3.1,*512,3072,2133,**500,1,**4.0,2.0,2366,**2331,***2694,*1.0,0,1,**0,0,**0,0,7391

8,16,**3.0,***4.0,*3072,8192,3200,**0,1,**0.0,4.0,5083,**5101,***5034,*0.0,0,0,**0,1,**0,1,15218

2,4,**1.2,***3.6,*1024,4096,2667,**0,1,**0.0,1.0,3255,**3237,***3542,*0.0,1,0,**0,0,**0,1,10034

4,8,**1.6,***4.2,*1024,6144,1333,**0,1,**2.0,1.0,3820,**3757,***3806,*1.0,0,1,**0,0,**0,1,11383

4,8,**2.1,***3.7,*2048,4096,2400,**0,1,**4.0,4.0,3670,**3719,***3861,*1.0,0,1,**0,0,**0,1,11250

4,8,**2.4,***4.1,*1024,8192,2666,**0,1,**4.0,0.0,3624,**2901,***4122,*1.0,0,1,**0,0,**0,1,10647

4,4,**2.1,***2.3,*512,4096,2400,**0,1,**0.0,4.0,3028,**3253,***3258,*0.0,1,0,**0,0,**0,1,9539

4,8,**1.6,***1.8,*1024,6144,2400,**0,1,**0.0,4.0,3460,**3652,***3757,*0.0,1,0,**0,0,**0,1,10869

4,8,**2.4,***4.1,*1024,8192,2666,**0,1,**4.0,0.125,4101,**4266,***4304,*1.0,0,1,**0,0,**0,1,12671

2,2,**2.2,***2.5,*1024,0,1866,**500,0,**0.0,0.125,1221,**1241,***1411,*0.0,1,0,**0,0,**0,0,3873

2,4,**2.0,***2.0,*512,3072,2133,**0,1,**2.0,0.0625,1910,**2043,***2071,*1.0,0,1,**0,0,**0,1,6024

4,8,**2.1,***3.7,*2048,4096,2400,**0,1,**0.0,2.0,2417,**2411,***2385,*0.0,1,0,**0,0,**0,1,7213

4,8,**1.3,***3.9,*2048,8192,3733,**0,1,**0.0,8.0,3772,**3825,***3864,*0.0,1,0,**0,0,**0,1,11461

4,8,**1.6,***3.4,*1024,6144,2400,**0,1,**0.0,0.128,3278,**2033,***3370,*0.0,1,0,**0,0,**0,1,8681

4,8,**2.6,***3.8,*2048,4096,3200,**0,1,**0.0,0.5,2776,**2776,***2809,*0.0,1,0,**0,0,**0,1,8361

2,4,**2.5,***3.1,*512,4096,2133,**1000,1,**2.0,0.125,2343,**2771,***2581,*1.0,0,1,**0,0,**0,1,7695

2,4,**1.7,***2.7,*256,3072,1600,**0,1,**0.0,1.0,2191,**2165,***2257,*0.0,0,0,**0,0,**1,1,6613

4,8,**1.0,***1.2,*512,6144,2667,**0,1,**2.0,1.9,2774,**3919,***3551,*1.0,0,1,**0,0,**0,1,10244