ГУАП

КАФЕДРА № 41

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cтарший преподаватель |  |  |  | В. В. Боженко |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| Регрессионный анализ данных 2024 по курсу: ВВЕДЕНИЕ В АНАЛИЗ ДАННЫХ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4217 |  |  |  | В. А. Милованов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

**Цель работы**

Изучение алгоритмов и методов регрессии на практике.

**Индивидуальное задание**

Вариант 9

Часть 1

x1: 3, 3, 6, 6, 7, 8, 9

y: 23.7, 24.8, 25.8, 27.9, 26.9, 25.2, 26.6

Часть 2:

x: 0, 1, 2, 3, 4, 5, 6

y: 15.5, 25.4, 36.4, 39.9, 43.3, 38.8, 49.1

**Ход работы**

Были импортированы следующие библиотеки: pandas для работы с датасетами, numpy для реализации массивов, matplotlib для построения графиков, элементы из sklearn для работы с моделями. Были созданы 2 массива x1 и y с данными для прогнозирования. Целевым признаком выступает y, а побочным x1 (см. рис. 1).

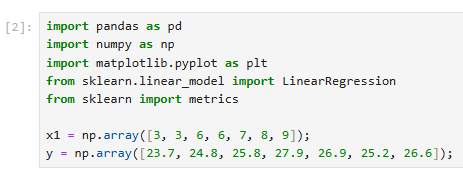


Рисунок 1 – Импорт библиотек

Была создана модель линейной регрессии с помощью импортированного ранее метода LinearRegression(). Использовались датафреймы X и Y, основанные на первичных данных, для обучения модели методом fit() (см. рис. 2).

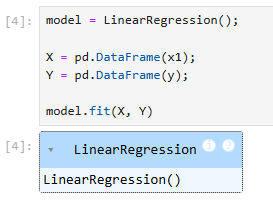


Рисунок 2 – Обучение модели

Модель была успешно обучена. Был реализован метод predict(), который прогнозирует значения y на базе информации из x1 и обученной модели. Метод ravel() приводит массив с результатами к одномерному виду (см. рис. 3).

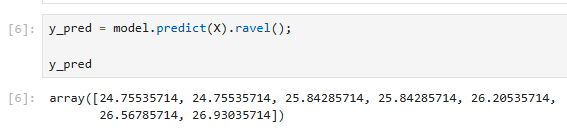


Рисунок 3 – Прогнозирование

Был создан и выведен датафрейм с истинными и предсказанными значениями, Actual и Predicted соответственно (см. рис. 4).

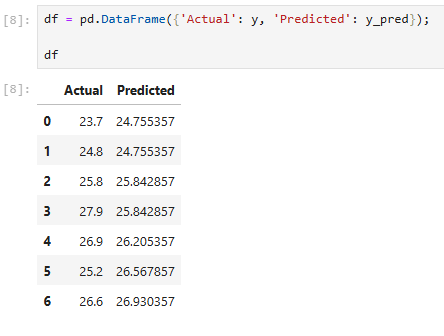


Рисунок 4 – Датафрейм с прогнозами

Заметно, что предсказания ориентировочно совпадают с первичными данными.

Были использованы методы расчета метрик из sklear, а точнее объект класса metrics. Рассчитывались средняя абсолютная ошибка, среднеквадратическая ошибка, корень из среднеквадратической ошибки и коэффициент детерминации. Были выведены результаты (см. рис. 5).

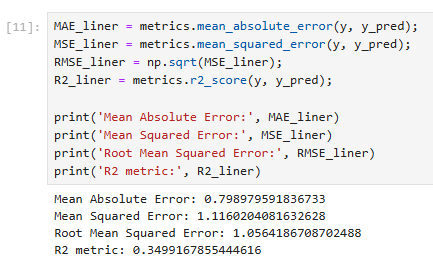


Рисунок 5 – Метрики первой модели

Малые значения ошибок объясняются небольшим объемом данных для прогнозирования. В свою очередь, низкий коэффициент детерминации обусловлен малым количеством информации для обучения.

Для вывода коэффициентов a и b использовались методы из sklear, а именно coef\_ и intercept\_ соответственно (см. рис. 6).

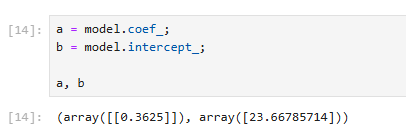


Рисунок 6 – Коэффициенты первой модели

Была произведена визуализация результатов, реализация точек начальных данных с использованием scatter() и линии регрессии с помощью plot() (см. рис. 7-8).

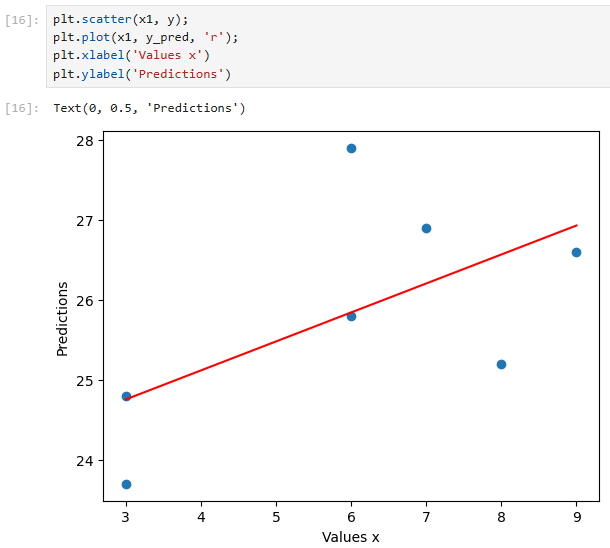


Рисунок 7 – Визуализация результатов обучения первой модели

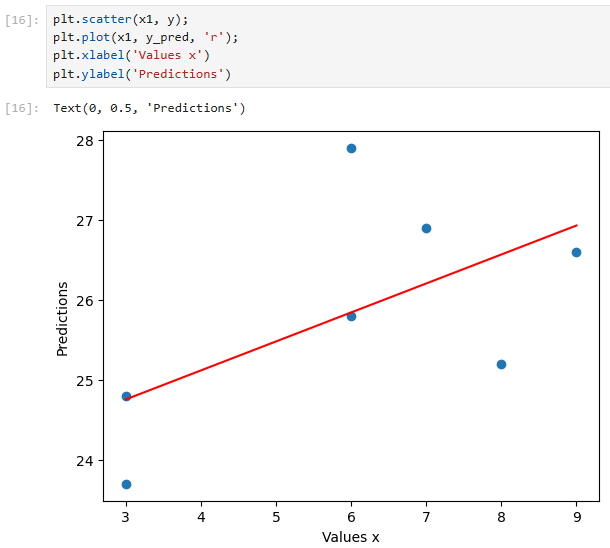


Рисунок 8 – График результатов

На графике заметны пересечения верных значений и динамика прогнозируемых.

Был создан график с отношением предсказанных значений к линии верных с помощью scatter() и plot() соответственно, были добавлены пояснения условных обозначений методом legend() (см. рис. 9).

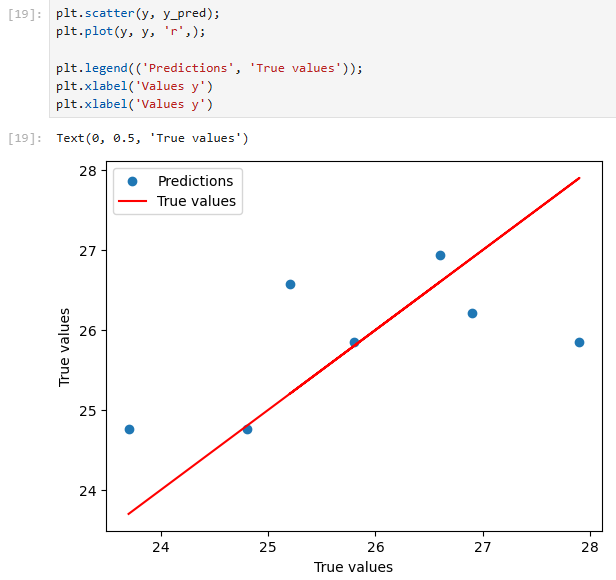


Рисунок 9 – Предсказанные и истинные величины первой модели

Был импортирован метод PolynomialFeatures для реализации модели полиномиальной регрессии. Инициализированы массивы x и y с данными для обучения и проверки модели. Степень полинома была установлена на 2 (см. рис. 10).

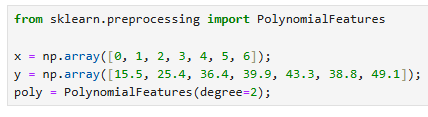


Рисунок 10 – Реализация данных для второй модели

Особенностью полиномиальной модели является трансформация побочного признака методом poly.fit\_transform(). Метод преобразует X в массив, содержащий результат возведения в степень исходных данных. Количество значений зависит от указанного параметра degree (см. рис. 11).

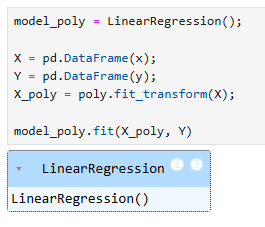


Рисунок 11 – Обучение второй модели

После обучения полиномиальной модели были сделаны и выведены следующие предсказания величины y аналогичным образом (см. рис. 12).

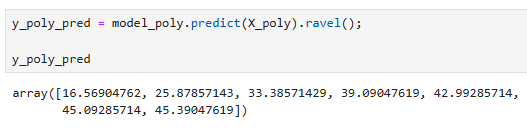


Рисунок 12 – Прогнозы второй модели

Были рассчитаны ранее используемые метрики (см. рис. 13).

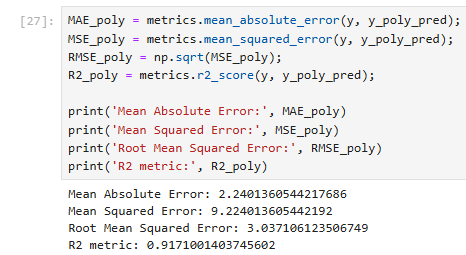


Рисунок 13 – Метрики второй модели

Была произведена визуализация полиномиальной регрессии. На графике указаны точки по первичным значениям и регрессионная линия (см. рис. 14).

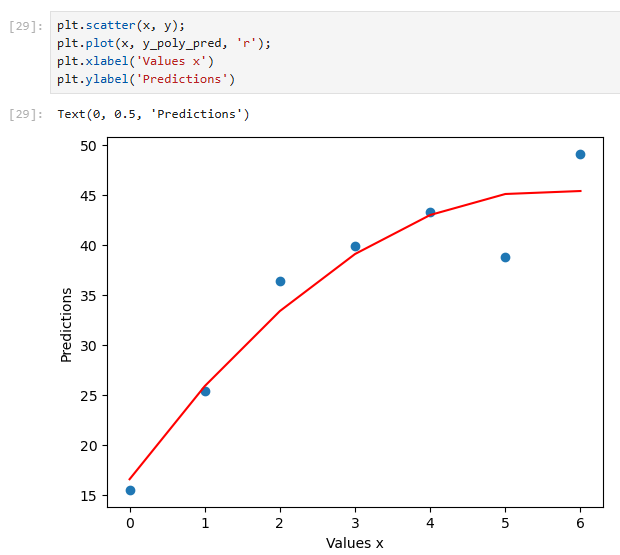


Рисунок 14 – Полиномиальная модель со степенью 2

Можно заметить изменения линии регрессии. Она стала более точно описывать закономерность распределения значений.

Аналогичным образом была обучена ещё одна полиномиальная модель с разницей в степени полинома, на этот раз равной 3. Производилось прогнозирование значений и визуализация результатов (см. рис. 15).

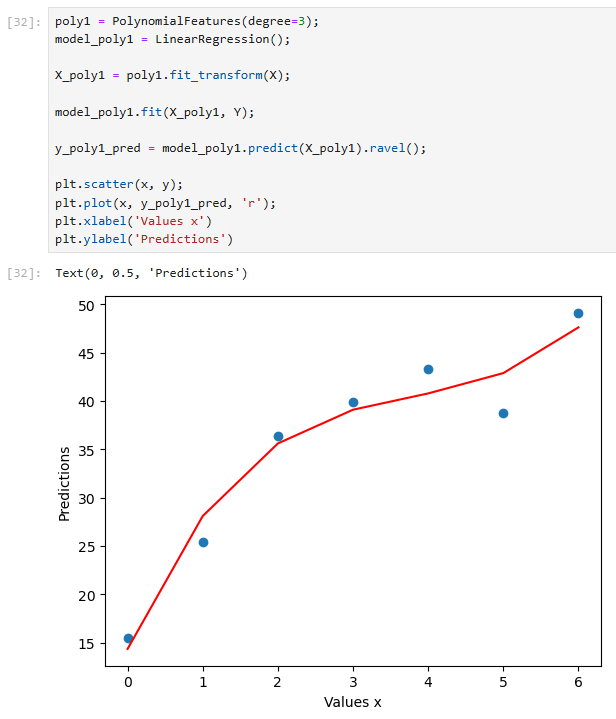


Рисунок 15 – Полиномиальная модель со степенью 3

Полиномиальная модель отличается от линейной тем, что способна моделировать нелинейные зависимости. Линейная регрессия описывает данные с помощью прямой линии, тогда как полиномиальная добавляет степени признаков, что позволяет описывать кривые зависимости. В данном случае регрессионная линия более точно отобразила закономерность между точками исходных значений. Параметр степени полинома определяет сложность модели: при низкой степени модель приближает простые кривые, а высокая степень позволяет описывать сложные зависимости, но может привести к переобучению, когда модель слишком точно подстраивается под шум. Оптимальный выбор степени важен для обеспечения баланса между точностью и обобщающей способностью модели.

Методом pandas был импортирован набор данных car\_price.csv, содержащий подробную информацию о различных характеристиках конкретных автомобилей и установленной на них цены (см. рис. 16).

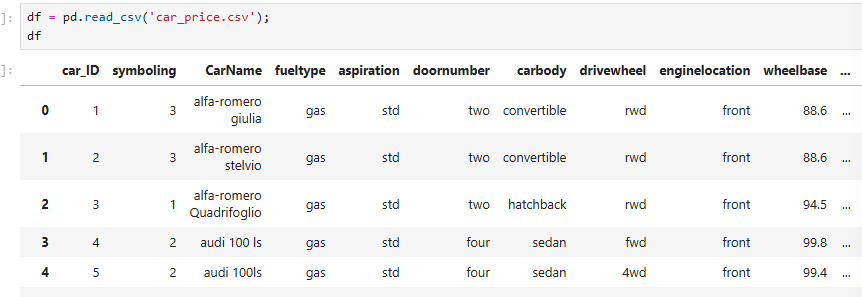


Рисунок 16 – Набор данных car\_price.csv

Была проанализирована информация по car\_price.csv в методических указаниях и строчки с данными об автомобилях в среде. В первую очередь был определен целевой признак, а именно цена (price). Этот признак был выбран, так как стоимость автомобиля складывается в зависимости от всех остальных его характеристик, особенно выделенных в рамках приведенной БД.

По значениям цены был построен boxplot с использованием методов boxplot() для реализации графика, title(), xlabel() для заголовков и grid() для настройки сетки (см. рис. 17).

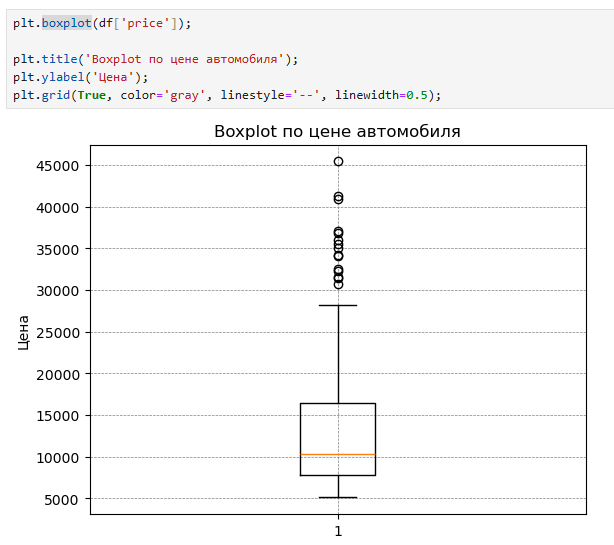


Рисунок 17 – Boxplot по цене

На графике заметны выскакивающие значения. Они были удалены с помощью ограничения верхнего порога значений по столбцу price до 30k (см. рис. 18).



Рисунок 18 – Удаление выбросов

Далее по значениям стоимости автомобиля была построена гистограмма с использованием аналогичных методов и hist() (см. рис. 19).

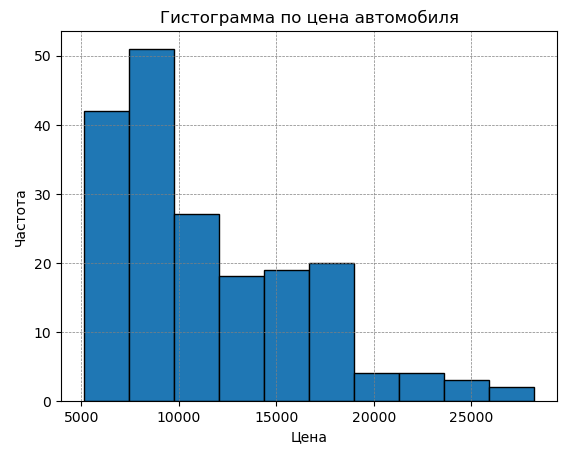


Рисунок 19 – Гистограмма по цена автомобиля

Для обучения регрессионной модели требуется выделить побочные признаки. Сразу отбросим категориальные величины. Обратим внимание на числовые признаки набора данных. Оценим размах величин по max и min по значениям в выводе метода describe() (см. рис. 20).

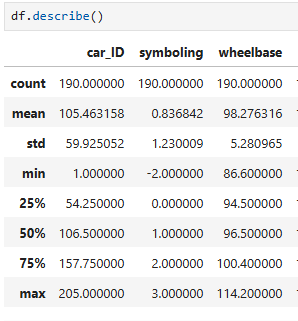


Рисунок 20 – Описание числовых данных

Следующим шагом отбросим коэффициенты и другие малые значения (symboling, boreratio, stroke , compressionratio), так как они более обобщены и по своей сути близки скорее к качественным признакам.

Для остальных величин построим матрицу рассеивания в отношении цен на автомобили. Реализуем задуманное с помощью библиотеки seaborn (см. рис. 21).

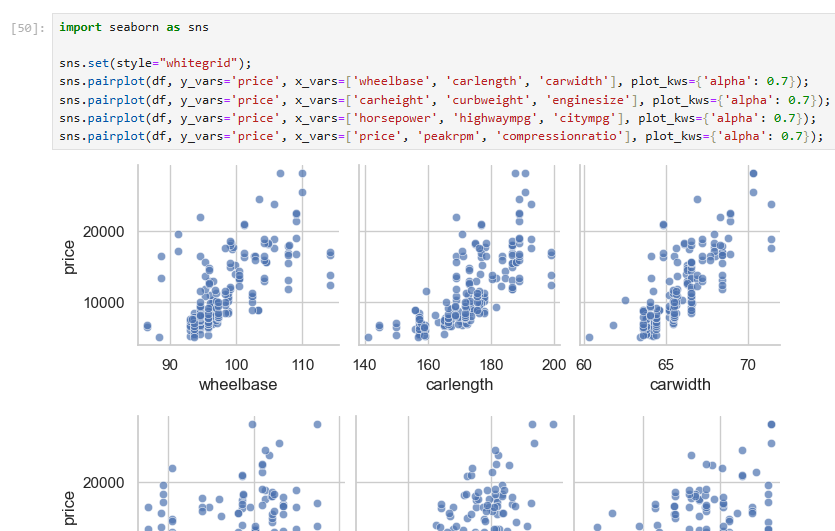


Рисунок 21 – Матрица рассеивания

По графикам видно, что наибольшая зависимость между ценой и конкретным признаком наблюдается у следующих величин: wheelbase, carlength, carwidth, curbweight, enginesize, horsepower. Остальные признаки имеют более хаотичную форму значений относительно стоимости, и предположения насчет compressionratio были подтверждены.

Для точности умозаключений были проверены значения коэффициентов корреляции между ценой и остальными числовыми признаками. Использовался метод corrwith c параметром numeric\_only (см. рис. 22).

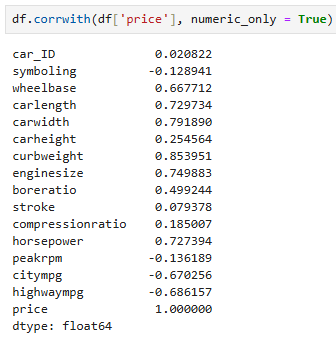


Рисунок 22 – Коэффициенты корреляции данных

Ранние предположения были подтверждены, поэтому для обучения модели в качестве побочных признаков были приведены: wheelbase, carlength, carwidth, curbweight, enginesize, horsepower.

Из библиотеки sklearn были импортированы методы StandardScaler для нормализации данных и train\_test\_split для разбиения данных на обучающую и тестовую выборки. Нужные столбцы для прогнозирования были выбраны и занесены в переменные X и Y. Нормализация применялась к побочным признакам X. Данные были разделены на X\_train, X\_valid, y\_train и y\_valid. Для обучения модели использовались X\_train и y\_train (см. рис. 23).



Рисунок 23 – Обучение третьей модели

Были сделаны предсказания относительно цены автомобиля. Качество прогнозирования было оценено с помощью средней абсолютной ошибки, среднеквадратической ошибки, корню из среднеквадратической ошибки и коэффициентом детерминации (см. рис. 24).

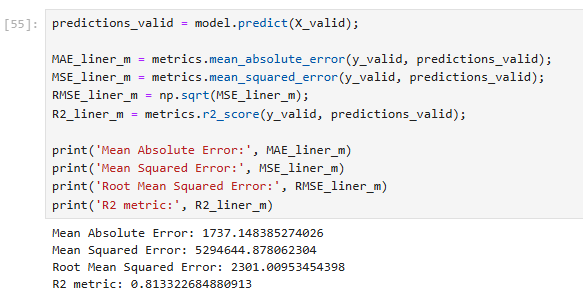


Рисунок 24 – Метрики третьей модели

Коэффициент детерминации близок к 0, а именно равен 0,81, что говорит о способности модели объяснить большую часть данных. Помимо этого, под другим метрикам заметно малое количество ошибок относительно такого объема данных. Таким образом, показатели созданной модели довольно приемлемы.

Был создан датафрейм из истинных и предсказанных значений, Actual и Predicted. Обновлена индексация. Используя метод head, были выведены несколько строк датафрейма (см. рис. 25).

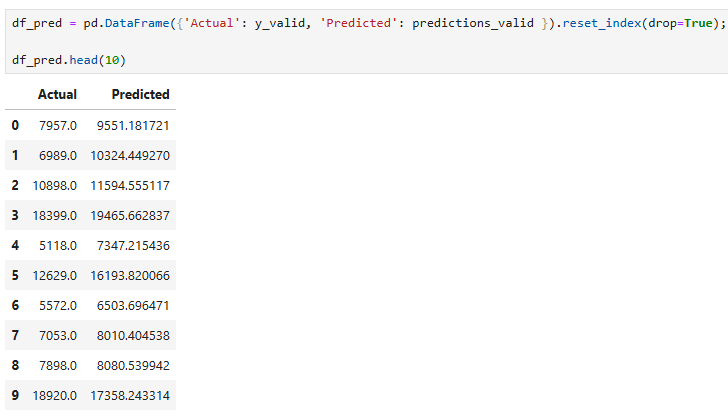


Рисунок 25 – Датафрейм с прогнозами третьей модели

Ещё один датафрейм был выведен. В нем показываются коэффициенты регрессии a и b в зависимости от побочного признака (см. рис. 26).

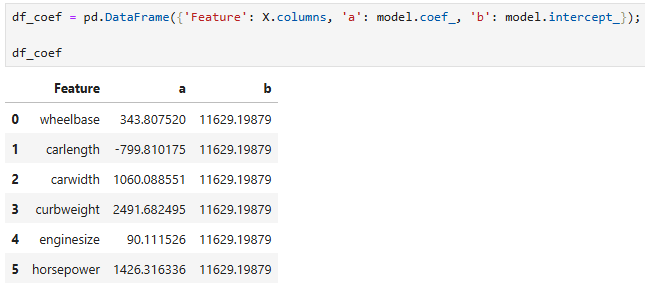


Рисунок 26 – Вывод коэффициентов по побочным признакам

Был построен аналогичный 1-й части задания график отображающий отношение верных и предсказанных значений. Были обозначены в легенде как Predictions и True values (см. рис. 28).

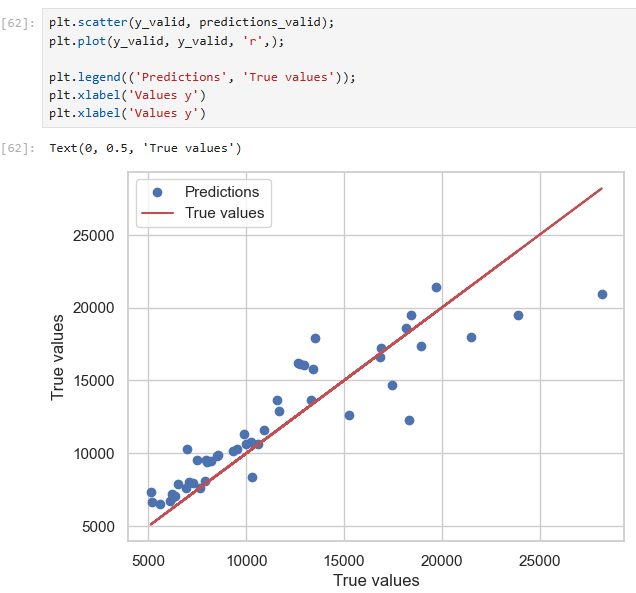


Рисунок 28 – Визуализация третьей модели

Дополнительно была обучена ещё одна регрессионная модель только в этом случае методом k-ближайших соседей. К-ближайших соседей (K-Nearest Neighbors или просто KNN) — алгоритм классификации и регрессии, основанный на гипотезе компактности, которая предполагает, что расположенные близко друг к другу объекты в пространстве признаков имеют схожие значения целевой переменной или принадлежат к одному классу. Использовался импортированный метод KNeighborsRegressor c параметром n\_neighbors=27. Такое значение параметра обусловлено его эффективностью по сравнению с остальными числовыми значениями. При значениях до 27 эффективность модели последовательно увеличивается, а после начинает стремительно падать (см. рис. 29).

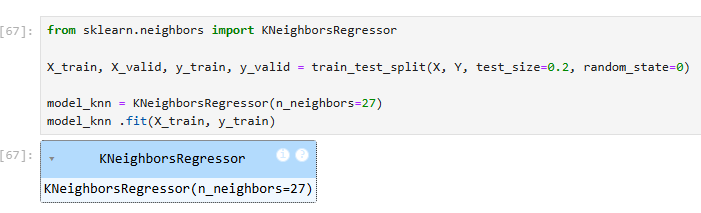


Рисунок 29 – Обучение модели методом k-ближайших соседей

Для сравнения качества нынешней модели в сравнении с предыдущей используются аналогичные метрики оценивания (см. рис. 30).

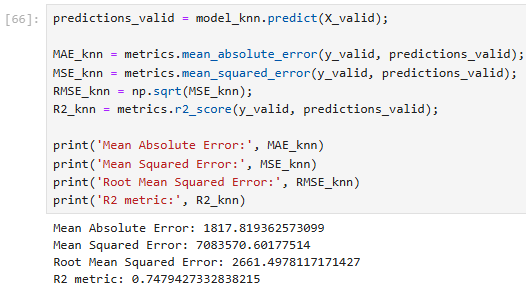


Рисунок 30 – Метрики модели k-ближайших соседей

Была проведена визуализация для более четкого понимания ситуации (см. рис. 31).

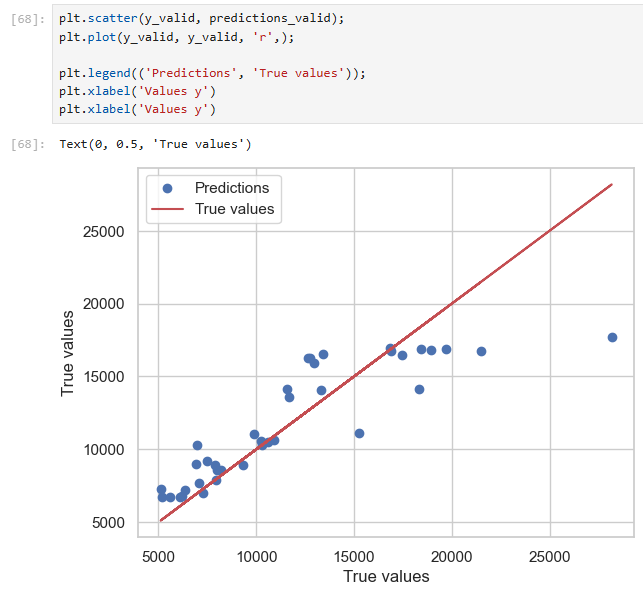


Рисунок 31 – Визуализация модели k-ближайших соседей

Можно заметить, что новая модель справляется хуже, чем старая. Меньшее качество работы модели k ближайших соседей (knn) по сравнению с линейной регрессией может быть связано с несколькими факторами. Линейная регрессия лучше справляется с линейными зависимостями и менее чувствительна к шуму в данных. Метод knn сильно зависит от параметра k, выбора метрики расстояния и чувствителен к выбросам, что может приводить к переобучению или недообучению. Кроме того, в условиях высокой размерности (проклятие размерности) эффективность knn падает, так как расстояния между точками становятся менее информативными. Также важно масштабирование данных для knn, так как разные масштабы признаков могут исказить результаты.

Ссылка на файл:

<https://github.com/Vvvvv55531/LR3.git>

**Вывод**

В ходе работы было изучено отличие между моделями линейной и полиномиальной регрессии, их особенности и применение в задачах анализа данных. Было разобрано, как простая линейная регрессия используется для моделирования линейных зависимостей, тогда как полиномиальная регрессия позволяет описывать сложные, нелинейные отношения между переменными с использованием степенных признаков. Стало понято, что выбор степени полинома играет ключевую роль: низкая степень подходит для простых зависимостей, тогда как высокая степень повышает гибкость модели, но может привести к переобучению. Подчеркнута важность баланса между сложностью модели и её способностью к обобщению, что требует внимательного подбора степени полинома. Было изучено использование метода k ближайших соседей и его особенности. Осознано, что knn чувствителен к шуму, выбору параметра k и масштабу данных, что делает нормализацию важной для корректной работы. Обнаружено, что метод может уступать линейной регрессии при наличии выраженной линейной зависимости. Изучено понятие "проклятие размерности", которое объясняет снижение эффективности метода knn при работе с высокоразмерными данными. В рамках работы были освоены построение и анализ различных графиков, включая scatter plot, boxplot и гистограммы, а также построение матриц рассеивания для оценки взаимосвязей между признаками. Стало понято, как улучшать внешний вид графиков, добавлять легенды и настраивать оси для более информативного отображения данных. Было изучено использование методов нормализации и масштабирования для подготовки данных. Изучено, как метод fit\_transform() помогает преобразовывать данные для применения в моделях, и как использовать PolynomialFeatures для создания полиномиальных признаков. Были исследованы способы решения ошибок, возникающих при работе с массивами, включая корректировку форматов данных и правильное использование методов reshape() и .item(). Было изучено, как работать с многомерными массивами и приводить их к нужной форме для дальнейшей обработки.