

Оглавление

[1.АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#_Toc184657542)

[1.1 Поиск набора данных 4](#_Toc184657543)

[1.2 Оценка загруженного набора данных 5](#_Toc184657544)

[1.3 Изучение и выбор подходящего вида моделей машинного обучения 12](#_Toc184657545)

[2. ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ 14](#_Toc184657546)

[2.1. Формирование обучающей и тестирующей выборок 14](#_Toc184657547)

[2.2. Создание моделей 14](#_Toc184657548)

[3. ТЕСТИРОВАНИЕ 17](#_Toc184657549)

[3.1. Проверка 17](#_Toc184657550)

[3.2. Улучшение модели 18](#_Toc184657551)

[ЗАКЛЮЧЕНИЕ 18](#_Toc184657552)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ 19](#_Toc184657553)

**ВВЕДЕНИЕ**

В качестве основы для выполнения курсовой работы был использован набор данных, содержащий большое количество подробной информации об автомобилях и их характеристиках.

Актуальность данной работы важна, так как в данный момент возрос спрос на автомобили в связи с сложностями транспортировки, а колебания цен на одну и туже модель, может быть, колоссальной.

**Цель**

Провести работы по построению ансамбля для нахождения ценны автомобиля на основании ранее предоставленных данных для обучения.

**Задачи**

1. Анализ предметной области
   1. Подбор подходящего набора данных
   2. Оценка отобранного набора данных
   3. Изучение и выбор подходящего вида моделей машинного обучения
2. Разработка, обучение модели
   1. Формирование обучающей и тестовой выборки
   2. Обучение модели
   3. Создание нужных моделей
3. Тестирование и отладка
   1. Проверка продуктивности
   2. Улучшение модели

Работа состоит из введения, трёх глав, заключения, библиографии. Объем работы составляет 22 страниц, объем библиографии содержит 12 источников.

# **1.АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

## **1.1 Поиск набора данных**

В результате поиска информации в сфере автомобилей и их характеристик был найден сборник данных, в формате csv, содержащий большое количество (76690 строчек) информации о выбранной тематике, в качестве перестраховки данные хранятся на облачном хранилище (ссылка для набора данных: https://drive.google.com/file/d/1hmDNBf86OBtjHELEcceO0nEUowYM7jEm/view?usp=sharing), на внешнем диске и на платформе GitHub (https://github.com/DanilaShanin/Coursework.git). Ниже приведены названия столбцов данного набора среди них находятся текстовые или смешанные числовые значения, целые числа и числа с плавающей точкой:

**Make –Имя производителя**

**Name – Название модели**

**Transmission – Вид трансмиссии**

**Engine Type – Наименование используемого вида топлива**

**Engine Capacity(CC) – Мощность**

**Mileage(kms) – Пробег**

**City – Город продажи**

**Year – Год выпуска**

**Price - Цена**

Для загрузки информационного набора используется команда

|  |
| --- |
| df = pd.read\_csv('/content/CarDF.csv') |

После успешного завершения процесса загрузки информации была выполнена проверка корректности данных. Результатом данной проверки стало то , что данные были загружены без каких-либо ошибок. Проблемы с кодировкой символов отсутствуют, что гарантирует правильное отображение всех текстовых элементов и символов. Кроме того, все разделители данных были идентифицированы и обработаны корректно, что обеспечивает целостность структуры информации. Эти результаты свидетельствуют о том, что загрузка прошла успешно Рисунок1.

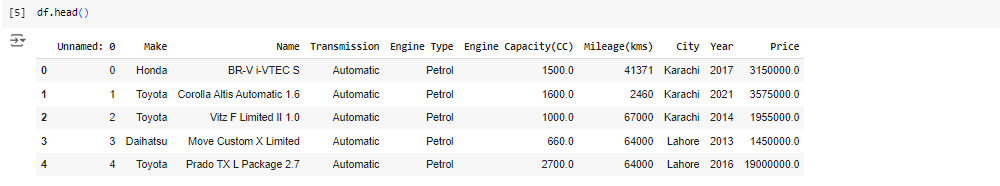


Рисунок 1 - Набор данных

## **1.2 Оценка загруженного набора данных**

После загрузки и проверки производился анализ полученного набора информации с использованием библиотеки matplotlib, которая позволяет выводить различные виды графиков.

Для начала проводилась проверка на наличие выбросов с помощью графика boxplot, а дальнейшая обработка проводилась с помощью RobustScaler.

|  |
| --- |
| sns.boxplot(data=df, x="Engine Capacity(CC)", ax=axs[0, 0])  sns.boxplot(data=df, x="Mileage(kms)", ax=axs[0, 1])  axs[0, 0].set\_title("Распределение объемов двигателей (CC)")  axs[0, 1].set\_title("Распределение пробегов (kms)")  for i in range(2):  for j in range(2):  if j != 0:  axs[i, j].get\_yaxis().set\_visible(False)  axs[i, j].set\_xticklabels(axs[i, j].get\_xticklabels(), rotation=45, ha='right') |

Результат выполнения кода представлен на Рисунке 2.

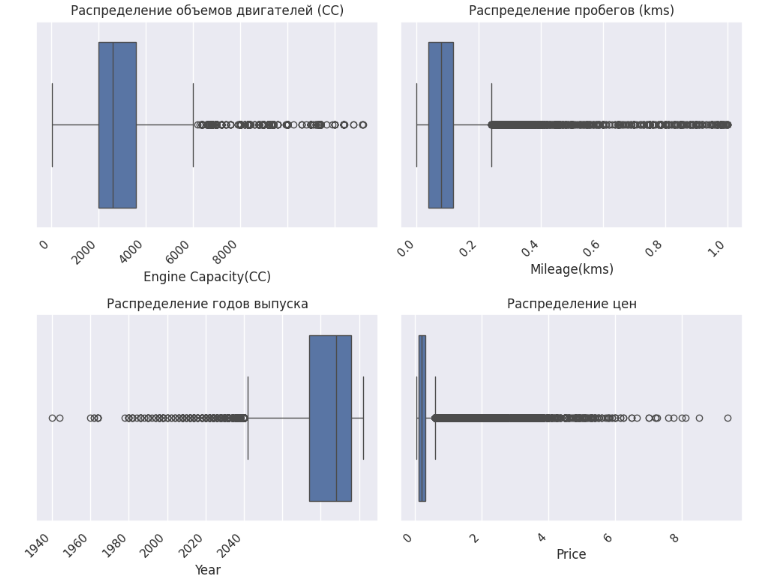


Рисунок 2 – Проверка выбросов

Для нахождения крупных рынков продажи авто, так же используется гистограмма с топом городов Рисунок 3.

|  |
| --- |
| cities = df.City.value\_counts()  sns.barplot(x=cities[:10].index,y=cities[:10].values)  plt.xlabel('Top 10 cities')  plt.ylabel('Cars For Sale')  plt.title('Top 10 cities',color = 'green',fontsize=20) |

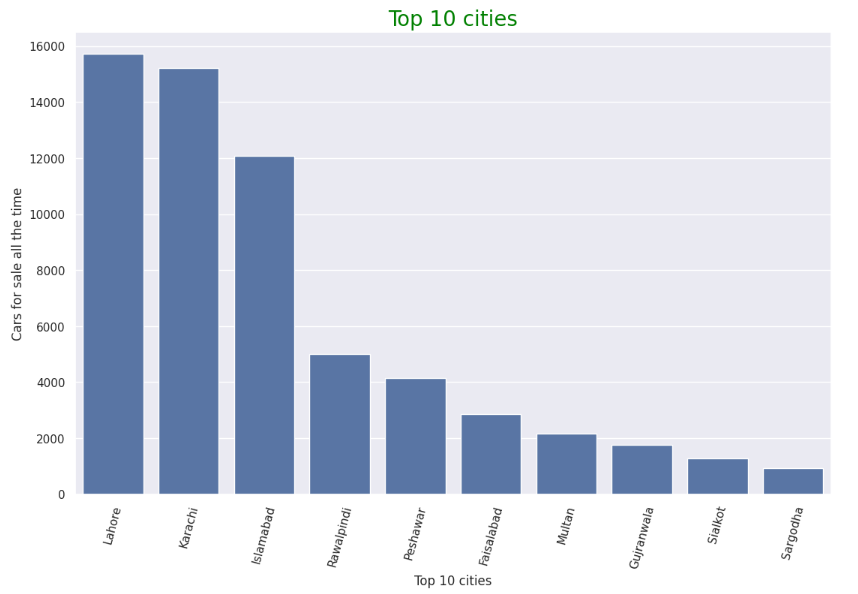


Рисунок 3 - Топ 10 городов по количеству авто

В итоге стало понятно, что самыми крупными городами где находится наибольшее количество авто выставленных на продажу являются: Лахор, Карачи и Исламабад являющаяся столицей Пакистана.

Просмотр распространённых ценовых категорий проводилась так же с помощью гистограммы Рисунок 4.

|  |
| --- |
| bins = [0,1000001,2000001,4000001,6000001,10000001,100000001]  labels = ['Below 1 million','1-2 million','2-4 million','4-6 million','6-10 million','10+ million']  df['Цены на автомобили и их количество'] = pd.cut(df['Price'],bins,labels = labels) |

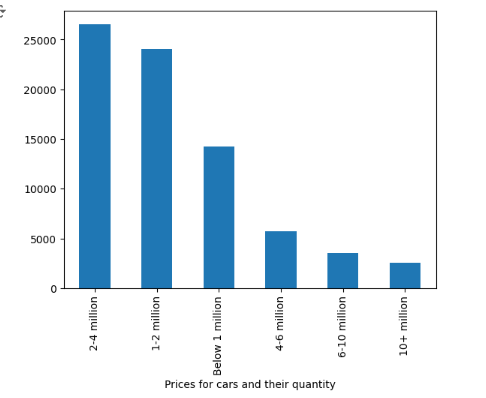


Рисунок 4 – Ценовые категории и количество авто

Наиболее популярной категорией являются цены от 2 до 4 млн, от 1 до 2 млн, а самой не популярной являются автомобили ценой свыше 10 миллионов.

Для отслеживания изменения средней стоимости автомобилей был построен график изображённый на Рисунке 5.

|  |
| --- |
| sns.set(style="darkgrid")  plt.figure(figsize=(10, 6))  ax = sns.lineplot(x='Year', y='Price', data=df)  ax.set\_title('Динамика изменения средней стоимости автомобилей')  ax.set\_xlabel('Год')  ax.set\_ylabel('Цена') |

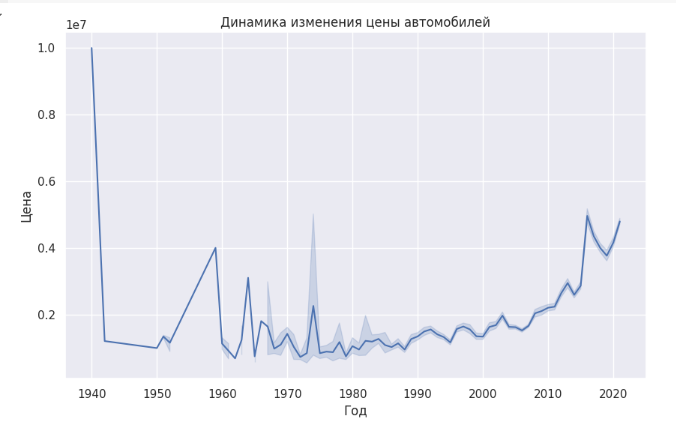


Рисунок 5 – Динамика изменения средней стоимости автомобилей

Круговая диаграмма помогает разобраться в возрасте автомобилей, находящихся в данном наборе информации Рисунок 6. Использовалось 4 временных промежутка для удобства понимания информации.

|  |
| --- |
| bins = [1940,2000,2010,2018,2021]  labels = ['1940-2000','2000-2010','2010-2018','2018-2021']  df['Car Age'] = pd.cut(df['Year'],bins,labels = labels) |

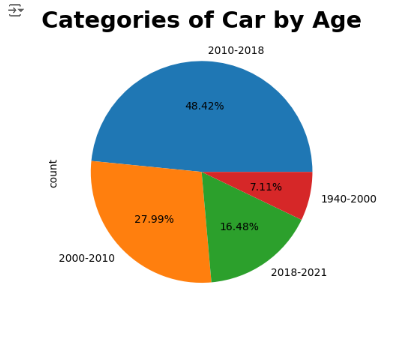


Рисунок 6 – Распределение авто по возрасту

После этого проводился анализ других характеристик автомобилей с помощью столбчатых диаграмм, узнать среднюю мощность автомобилей с разными коробками передач (автоматическая или механическая) и в разные временные промежутки Рисунок 7. Рассматривая данный график видно, что наиболее популярной трансмиссией является автоматическая.

|  |
| --- |
| sns.barplot(x=df['Car\_Age'],y=df['Engine Capacity(CC)'],hue=df['Transmission'])  plt.show() |

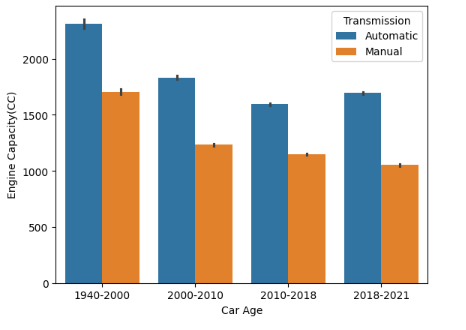


Рисунок 7 – Различие мощности и трансмиссий

Ценообразование транспорта с разными коробками передач проводилась так же с помощью столбчатых диаграмм Рисунок 8. Благодаря этому становится понятно, что цена на автомобили с автоматической коробкой передач выше не зависимо от возраста авто и возрастает с каждым поколением.

|  |
| --- |
| sns.barplot(x=df['Car Age'],y=df['Price'],hue=df['Transmission'])  plt.show() |

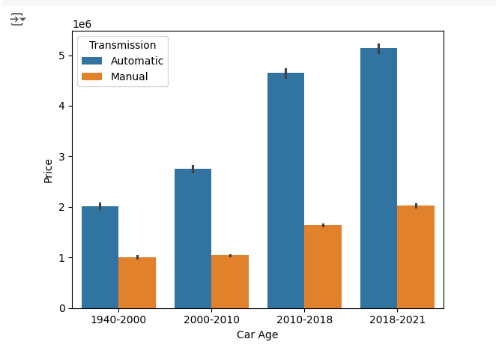


Рисунок 8 – Цены авто в зависимости от коробки передач

Просмотр распределенных данных выполняется с помощью: Гистограммы (Представление данных в рамках непрерывного интервала или ограниченного временного промежутка), Боксплота (Отображение статистических показателей: медиана, минимум и максимум, а так же выбросов), Точечной карты (Воспроизведение географических показателей), Графика плотности (Отображает более гладкое распределение за счёт сглаживания изменений), а для выявления зависимостей используются: Корреляционный анализ (Проверка степени линейной зависимости между несколькими переменными), Регрессионный анализ (методы, оценивающие связь между несколькими переменными с помощью математической модели, часто используемый в прогнозировании и поиске закономерностей), Деревья решений и случайные леса (алгоритмы машинного обучения, которые автоматически определяют зависимости) Рисунок 9.

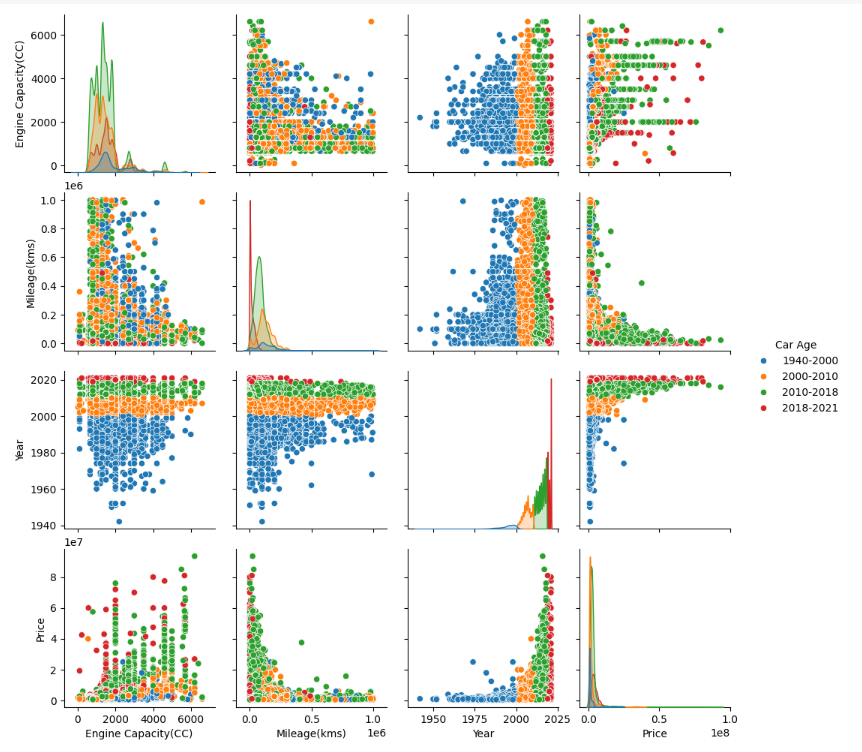
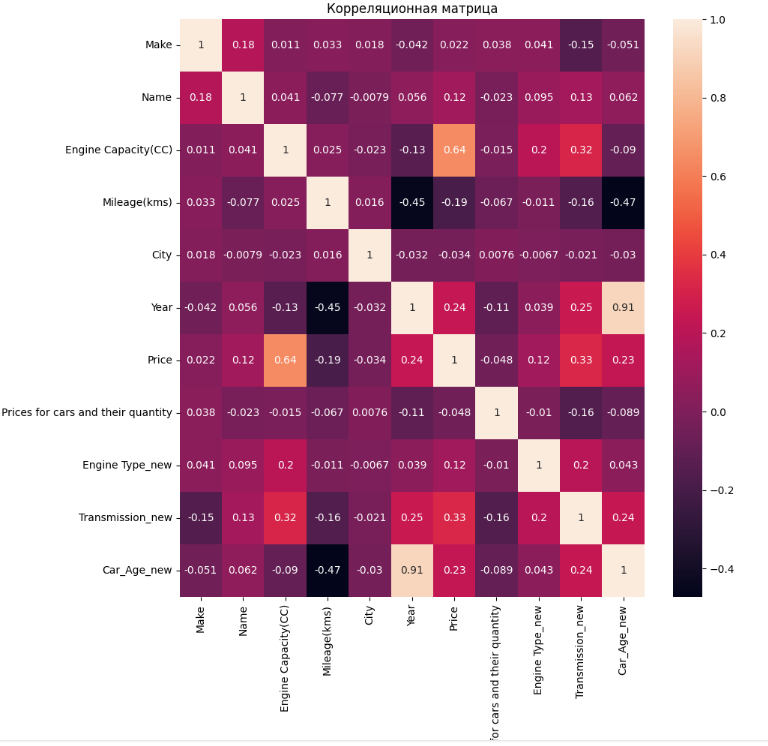


Рисунок 9 – Парный график зависимости

Так как использовалось много текстовых значений, их нужно перевести в цело числовые значения с помощью команды:

|  |
| --- |
| ordinal\_map = {'CNG':1,  'LPG':2,  'Petrol':3,  'Diesel':4,  'Hybrid':5,}  df['Engine Type\_new'] = df.Engine\_Type.map(ordinal\_map)  df=df.drop(['Engine\_Type'],axis=1)  df.head() |

После преобразования текстовых значений в числовые значений была построена Корреляционная матрица Рисунок 10.

  
Рисунок 10 - Корреляционная матрица

Исходя из коэффициентов корреляционной матрицы, наиболее коррелирующими признаками (фичами) являются: Engine Capacity(CC), Mileage(kms), Year, Engine Type\_new, Transmission\_new.

## **1.3 Изучение и выбор подходящего вида моделей машинного обучения**

Перед началом работ с ансамблевыми методами нужно понять их назначение. Ансамбли помогают объединить несколько моделей, на основе нужного алгоритма, для улучшения точности предсказаний по сравнению с одиночной моделью. Часто используемыми методами являются Рисунок 11.

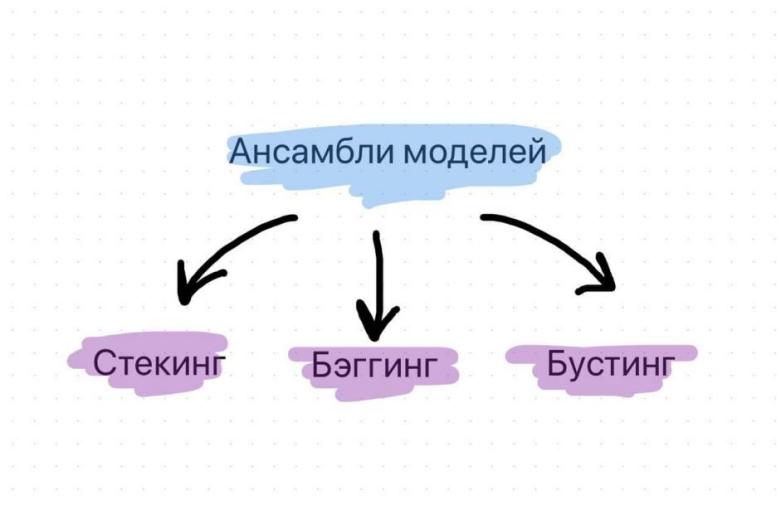


Рисунок 11 – Ансамбли моделей

Стекинг – это метод ансамблевого обучения, который используется для улучшения точности предсказаний моделей машинного обучения путём объединения нескольких базовых моделей в одну более сложную модель. Основной концепцией стекинга является то, что используются прогнозы различных моделей как входные данные для другой модели, называемой метамоделью, которая обучается на этих прогнозах и делает окончательное предсказание.

Бэтинг - так же рассматриваются отдельные модели, обучение которых проводится параллельно и независимо друг от друга, и итоге их значения усредняются. Наиболее известным примером является случайный лес.

Бустинг - в данном случае обучение проводится последовательно, каждая следующая модель должна быть лучше и исправлять ошибки предыдущих, одним из примеров является градиентный бустинг.

# **2. ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ**

## **2.1. Формирование обучающей и тестирующей выборок**

Перед началом работ нужно удалить выбросы с помощью RobustaScaler, принцип работы которого основан на использовании ключевых статистических показателей: медиана (Me)- значение разделяющее упорядоченные данные пополам , первый(Q1)- делит нижнюю половину данных пополам, точнее говоря 25% наблюдений будут меньше Q1 третий квартили(Q3)-как и Q1 делит значения пополам, но только в верхней части иначе 75% наблюдений будут меньше Q3 и межквартильный размах (IQR) - значение разницы между первым и третьим квартилями(*IQR*=*Q*3−*Q*1).

Для удаления выбросов определяется два диапазона: верхний рассчитывается с помощью формулы Q3+1.5\*IQR, а нижний Q1-1.5\*IQR. Значения, которые выходят за рамки диапазона считаются выбросами и масштабируются для уменьшения воздействия их на общий набор данных.

|  |
| --- |
| numerical\_features = ['Engine Capacity(CC)', 'Mileage(kms)', 'Year', 'Engine Type\_new', 'Transmission\_new', 'Price']  scaler = RobustScaler()  scaled\_data = scaler.fit\_transform(df[numerical\_features])  y = df['Price']  X = scaled\_data  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42) |

Определение числовых признаков и разделение данных

Таким образом RobustaScaler масштабирует числовые признаки и возвращает их обратно в DataSet.

## **2.2. Создание моделей**

На основании проведённого анализа были выделены подходящие модели: RandomForestRegressor - модель машинного обучения для прогнозирования значений с помощью объединения предсказаний нескольких деревьев решений; HistGradientBoostingRegressor(HGB) (модель использующая градиентный бустинг для улучшения результативности в задачах регрессии), AdaBoostRegressor(ABR) (оценщик который подгоняет регрессор на исходный набор данных, а затем подгоняет его копии на том же наборе), BaggingRegressor(BR) (метод регрессии базового уровня по случайным подвыборкам), ExtraTreesRegressor(ETR) (мета оценщик обучает рандомизируемые деревья решений) и StackingRegressor для обученния нескольких независимых моделей.

|  |
| --- |
| bagging\_model = BaggingRegressor(  n\_estimators=100,  max\_samples=0.5,  bootstrap=True,  oob\_score=True,  random\_state=42) |

Создание модели BaggingRegressor

Для StackingRegressor использовалось несколько вышеперечисленных моделей таких как RF, HGB, ABR.

|  |
| --- |
| stacking\_models = [  ('RF', RandomForestRegressor(  n\_estimators=100,  random\_state=42,  max\_depth=None,  min\_samples\_split=2)),  ('HGB', HistGradientBoostingRegressor(  max\_iter=25,  random\_state=42,  max\_leaf\_nodes=25,  learning\_rate=0.03,  l2\_regularization=0.001,  early\_stopping=True,  validation\_fraction=0.1,  tol=1e-4)),  ('ABR', AdaBoostRegressor(  n\_estimators=100,  learning\_rate=0.01,  random\_state=42 ))] |

Использование нескольких моделей для StackingRegressor

Для каждой выполняемой модели использовалась кросс-валидация -это операция по оценке качества работы, помогающий избежать переобучения и получить точную оценку. Основная идея заключается в том, чтобы разделить данные на несколько частей и поочерёдно использовать каждую как тестовый набор, а остальные как обучающий.

Основными преимуществами являются устойчивость к переобучению, использование всего набора информации и точность оценки.

|  |
| --- |
| cv\_scores = cross\_val\_score(stacking\_regressor, X\_train, y\_train, cv=5, scoring='neg\_mean\_absolute\_error')  cv\_mae = -1 \* np.mean(cv\_scores) |

Применение крос-валидации

**2.3. Обучение моделей**

Обучение проводится заранее подготовленных выборках

|  |
| --- |
| rf\_model.fit(X\_train, y\_train)  hgb\_model.fit(X\_train, y\_train)  adb\_model.fit(X\_train, y\_train) |

Обучение трёх моделей

Выполнение предсказания осуществляется с помощью .predict (это метод для выполнения предсказаний на основании массива, в результате выполнения получается массив с предсказаниями) на тестовой выборке с помощью данной команды

|  |
| --- |
| bagging\_preds = bagging\_model.predict(X\_test) |

Выполнение прогнозов на тестовой выборке

# **3. ТЕСТИРОВАНИЕ**

## **3.1. Проверка**

Оценка средней абсолютной ошибки определяется с помощью MAE – метрика для оценки точности модели регрессии.

mae\_stacking — это расчёт средней абсолютной ошибки (MAE) между значениями целевой переменной и предсказанными значениями.

MAE BaggingRegressor - вывод значений BaggingRegressor

Средний MAE по кросс-валидации - выводит среднее значение MAE

Обучение BaggingRegressor заняло - отображает время затраченное на обучение

|  |
| --- |
| mae\_stacking = mean\_absolute\_error(y\_test, stacking\_preds)  print(f'MAE BaggingRegressor: {mae\_bagging:.2f}')  print(f'Средний MAE по кросс-валидации: {cv\_mae:.2f}')  print(f'Обучение BaggingRegressor заняло {bagging\_training\_time:.2f} секунд') |

Вывод результатов модели BaggingRegressor

Результаты всех моделей представлены ниже Рисунок 12.

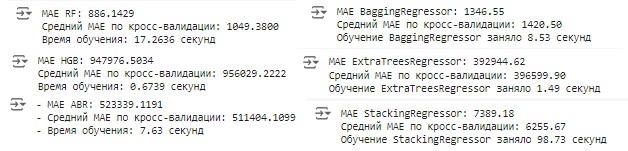


Рисунок 12 - Результаты моделей

## **3.2. Улучшение модели**

Подбор подходящих гиперпараметров проводился для повышения обобщающей способности сети и для получения наилучшего результата. В результате наиболее подходящими параметрами для модели RandomForestRegressor являются:

|  |
| --- |
| n\_estimators=100,  random\_state=42,  max\_depth=None,  min\_samples\_split=2) |

Параметры с наилучшим показателем

При данных параметрах модель смогла хорошо обучиться и уменьшить ошибку по сравнению с предыдущими моделями .Результат представлен на Рисунке 13.



Рисунок 13 -Результат модели RandomForestRegressor

# **ЗАКЛЮЧЕНИЕ**

Во время выполнения курсовой работы выполнялись работы по изучению, проектированию, и тестированию ансамблевых моделей машинного обучения. Данные модели позволяют спрогнозировать цену автомобиля на основании информационной выборки.

В результате работы, получен хороший фундамент для дальнейшей разработки программ для аналитики формирования цен на автомобили с разными конфигурациями.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ**

1. Чио, К. Машинное обучение и безопасность : руководство / К. Чио, Д. Фримэн ; перевод с английского А. В. Снастина. — Москва : ДМК Пресс, 2020. — 388 с. — ISBN 978-5-97060-713-8. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/131707 (дата обращения: 07.11.2024). — Режим доступа: для авториз. пользователей.
2. 50 оттенков matplotlib // Хабр URL: https://habr.com/ru/articles/468295/ (дата обращения: 08.12.2024).
3. RandomForestRegressor // Scikit Learn URL: https://scikit-learn.org/dev/modules/generated/sklearn.ensemble.RandomForestRegressor.html (дата обращения: 08.12.2024).
4. HistGradientBoostingRegressor // Scikit Learn URL: https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.HistGradientBoostingRegressor.html (дата обращения: 08.12.2024).
5. AdaBoostRegressor // Scikit Learn URL: https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.AdaBoostRegressor.html (дата обращения: 08.12.2024).
6. BaggingRegressor // Scikit Learn URL: https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.BaggingRegressor.html (дата обращения: 08.12.2024).
7. ExtraTreesRegressor // Scikit Learn URL: https://scikit-learn.org/dev/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html (дата обращения: 08.12.2024).
8. StackingRegressor // Scikit Learn URL: https://scikit-learn.org/dev/modules/generated/sklearn.ensemble.StackingRegressor.html (дата обращения: 08.12.2024).
9. RobustaScaler // Scikit Learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html (дата обращения: 08.12.2024).
10. mean\_absolute\_error // Scikit Learn URL: https://scikit-learn.org/1.5/modules/generated/sklearn.metrics.mean\_absolute\_error.html (дата обращения: 08.12.2024).
11. cross\_val\_score // Scikit Learn URL: https://scikit-learn.org/stable/modules/generated/sklearn.model\_selection.cross\_val\_score.html (дата обращения: 08.12.2024).
12. Титов, А. Н. Визуализация данных в Python. Работа с библиотекой Matplotlib : учебно-методическое пособие / А. Н. Титов, Р. Ф. Тазиева. — Казань : КНИТУ, 2022. — 92 с. — ISBN 978-5-7882-3176-1. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/331025 (дата обращения: 08.12.2024). — Режим доступа: для авториз. пользователей.