|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ***

***НА ТЕМУ:***

***\_\_\_\_\_\_« Регрессионные модели предсказания\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_стоимости домов в Калифорнии »\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_ИУ5-64Б\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_Б.О. Соколов\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** **\_\_\_\_****Ю.Е. Гапанюк\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2024 г.***Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_ИУ5\_\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_В.И. Терехов\_\_

(И.О.Фамилия)

«\_07\_» \_\_\_\_февраля\_\_\_\_\_\_\_ 2024 г.

**ЗАДАНИЕ**

**на выполнение научно-исследовательской работы**

по теме \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_ИУ5-64Б\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Соколов Борис Олегович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ИССЛЕДОВАТЕЛЬСКАЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_КАФЕДРА\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения НИР: 25% к \_\_\_\_ нед., 50% к \_\_\_\_ нед., 75% к \_\_\_ нед., 100% к \_\_\_\_ нед.

***Техническое задание*** \_\_\_\_\_Найти набор данных, провести его разведочный и корреляционный анализ, разработать и обучить 5 различных регрессионных моделей, после чего настроить их гиперпараметры и сравнить настроенные модели с базовыми на основе 3 выбранных метрик качества.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Оформление научно-исследовательской работы:***

Расчетно-пояснительная записка на \_\_25\_\_ листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_07\_» \_\_\_\_февраля\_\_\_\_\_\_\_ 2024 г.

**Руководитель НИР**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_Ю.Е. Гапанюк\_\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Б.О. Соколов\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

**Содержание:**

Введение 4

Описание исследования 5

Заключение и выводы 22

Список литературы 25

**Введение**

Прогнозирование стоимости жилья является ключевым аспектом в сфере недвижимости, особенно в таком разнообразном и экономически активном регионе, как Калифорния. Этот штат привлекает внимание не только из-за своего благоприятного климата и обилия природных красот, но и благодаря статусу центра инноваций и технологического прогресса. Силиконовая долина, Голливуд, мировые университеты – все это делает Калифорнию местом, где концентрируются финансовые потоки и культурная жизнь.

Стоимость жилья в Калифорнии подвержена колебаниям, которые зависят от множества факторов: экономических индикаторов, демографических тенденций, политических решений и даже экологических изменений. Точные прогнозы цен на недвижимость позволяют инвесторам, застройщикам и потенциальным домовладельцам принимать обоснованные решения, минимизируя риски и оптимизируя инвестиции.

Для государственных органов прогнозирование стоимости жилья важно для планирования налоговых поступлений, разработки социальных программ и управления градостроительными проектами. Это также помогает в анализе и предотвращении потенциальных экономических и социальных кризисов, связанных с жилищным рынком.

В контексте индивидуальных потребителей, знание тенденций стоимости жилья может помочь в планировании личных финансов, выборе момента для покупки или продажи недвижимости, а также в оценке стоимости собственного имущества.

Таким образом, прогнозирование стоимости домов в Калифорнии – это не просто академическое упражнение, а практическая необходимость, которая затрагивает экономическое благополучие и качество жизни многих людей. Использование передовых аналитических методов и алгоритмов машинного обучения позволяет с высокой точностью оценивать будущие изменения на рынке, что делает эту задачу особенно актуальной в современных условиях.

**Описание исследования:**

Подключим всё необходимое

import pandas as pd  
import numpy as np  
from sklearn.feature\_selection import VarianceThreshold  
from sklearn.decomposition import PCA  
import matplotlib.pyplot as plt  
import seaborn as sns  
from sklearn.impute import SimpleImputer  
from sklearn.compose import ColumnTransformer  
from sklearn.pipeline import Pipeline  
from sklearn.preprocessing import StandardScaler, OneHotEncoder  
from sklearn.metrics import r2\_score, mean\_absolute\_error, mean\_squared\_error  
from sklearn.tree import DecisionTreeRegressor  
from sklearn.svm import SVR  
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor  
from sklearn.neighbors import KNeighborsRegressor  
from sklearn.model\_selection import train\_test\_split  
from sklearn.model\_selection import GridSearchCV

data = pd.read\_csv('housing.csv', sep=",")

data.head()

longitude latitude housing\_median\_age total\_rooms total\_bedrooms \  
0 -122.23 37.88 41.0 880.0 129.0   
1 -122.22 37.86 21.0 7099.0 1106.0   
2 -122.24 37.85 52.0 1467.0 190.0   
3 -122.25 37.85 52.0 1274.0 235.0   
4 -122.25 37.85 52.0 1627.0 280.0   
  
 population households median\_income median\_house\_value ocean\_proximity   
0 322.0 126.0 8.3252 452600.0 NEAR BAY   
1 2401.0 1138.0 8.3014 358500.0 NEAR BAY   
2 496.0 177.0 7.2574 352100.0 NEAR BAY   
3 558.0 219.0 5.6431 341300.0 NEAR BAY   
4 565.0 259.0 3.8462 342200.0 NEAR BAY

data.shape

(20640, 10)

data.dtypes

longitude float64  
latitude float64  
housing\_median\_age float64  
total\_rooms float64  
total\_bedrooms float64  
population float64  
households float64  
median\_income float64  
median\_house\_value float64  
ocean\_proximity object  
dtype: object

Проведем разведку данных

Проверим есть ли пропущенные значения

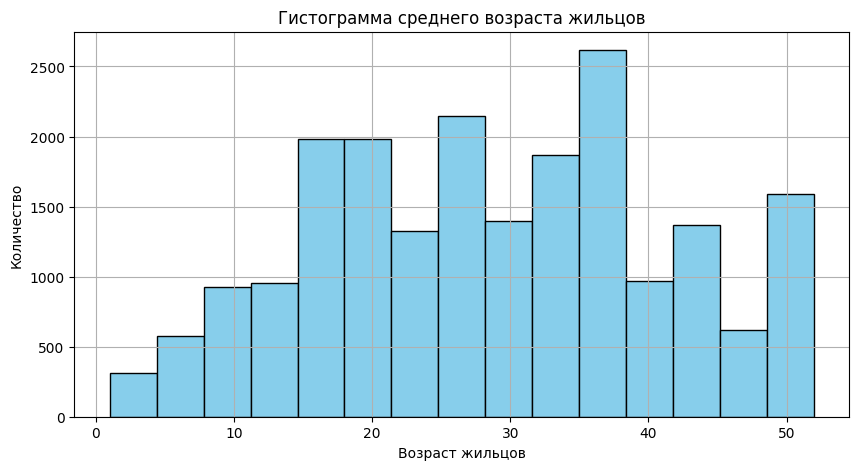
data.isnull().sum()

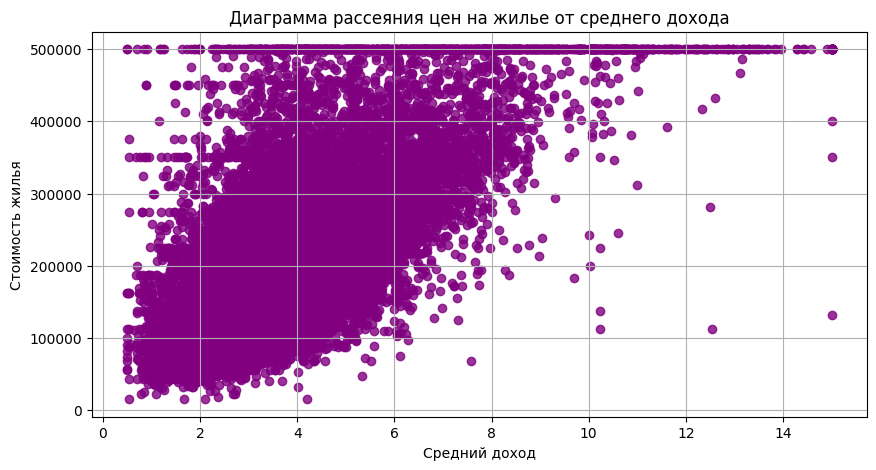
longitude 0  
latitude 0  
housing\_median\_age 0  
total\_rooms 0  
total\_bedrooms 207  
population 0  
households 0  
median\_income 0  
median\_house\_value 0  
ocean\_proximity 0  
dtype: int64

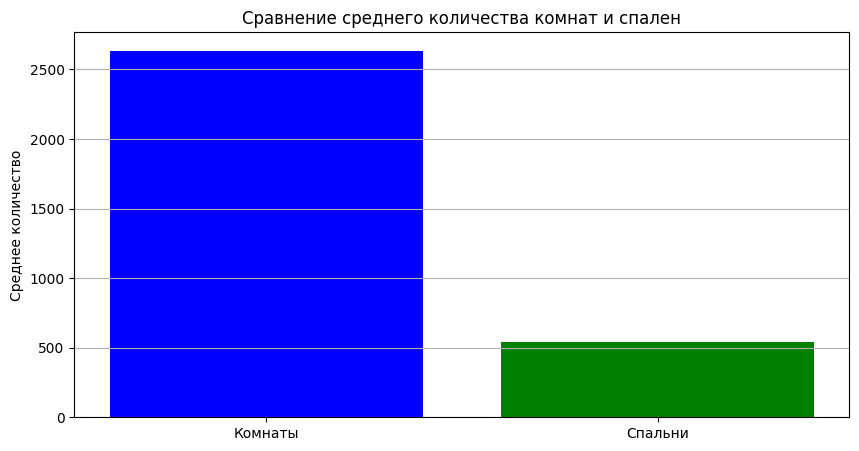
Заполним средним значением

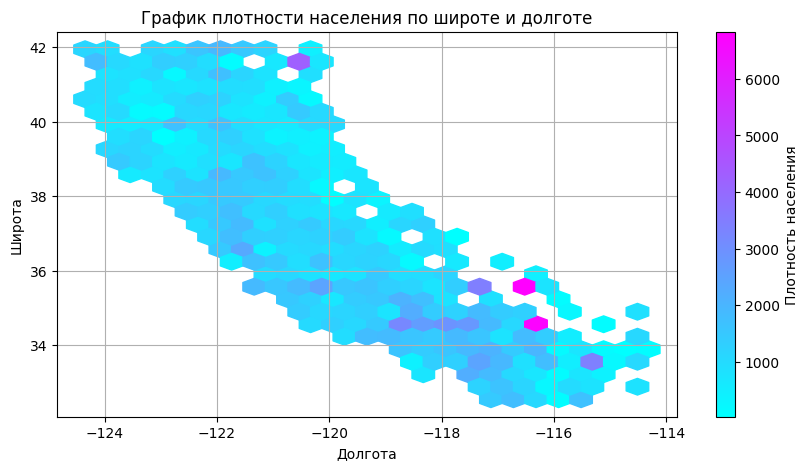
mean\_total\_bedrooms = data['total\_bedrooms'].mean()  
data['total\_bedrooms'].fillna(mean\_total\_bedrooms, inplace=True)

import matplotlib.pyplot as plt  
import pandas as pd  
  
  
# Гистограмма возраста жилья  
plt.figure(figsize=(10, 5))  
plt.hist(data['housing\_median\_age'], bins=15, color='skyblue', edgecolor='black')  
plt.title('Гистограмма среднего возраста жильцов')  
plt.xlabel('Возраст жильцов')  
plt.ylabel('Количество')  
plt.grid(True)  
plt.show()  
  
# Диаграмма рассеяния цен на жилье от медианного дохода  
plt.figure(figsize=(10, 5))  
plt.scatter(data['median\_income'], data['median\_house\_value'], alpha=0.8, c='purple')  
plt.title('Диаграмма рассеяния цен на жилье от среднего дохода')  
plt.xlabel('Средний доход')  
plt.ylabel('Стоимость жилья')  
plt.grid(True)  
plt.show()  
  
# Столбчатая диаграмма для сравнения среднего количества комнат и спален  
rooms = data['total\_rooms'].mean()  
bedrooms = data['total\_bedrooms'].mean()  
  
plt.figure(figsize=(10, 5))  
plt.bar(['Комнаты', 'Спальни'], [rooms, bedrooms], color=['blue', 'green'])  
plt.title('Сравнение среднего количества комнат и спален')  
plt.ylabel('Среднее количество')  
plt.grid(True, axis='y')  
plt.show()  
  
# График плотности населения по широте и долготе  
plt.figure(figsize=(10, 5))  
plt.hexbin(data['longitude'], data['latitude'], C=data['population'], gridsize=25, cmap='cool')  
plt.colorbar(label='Плотность населения')  
plt.title('График плотности населения по широте и долготе')  
plt.xlabel('Долгота')  
plt.ylabel('Широта')  
plt.grid(True)  
plt.show()

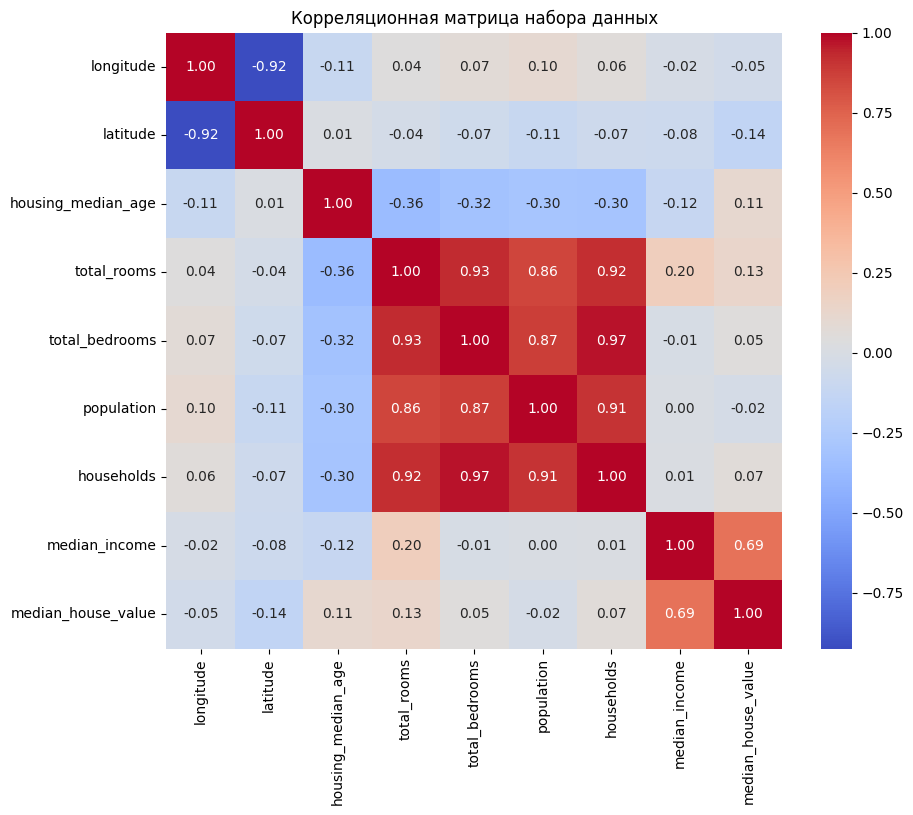








# Исключение нечисловых столбцов  
numeric\_data = data.select\_dtypes(include=[np.number])  
  
# Вычисление корреляционной матрицы для числовых столбцов  
corr\_matrix = numeric\_data.corr()  
  
# Создание тепловой карты корреляционной матрицы  
plt.figure(figsize=(10, 8))  
sns.heatmap(corr\_matrix, annot=True, fmt=".2f", cmap='coolwarm', square=True)  
plt.title('Корреляционная матрица набора данных')  
plt.show()



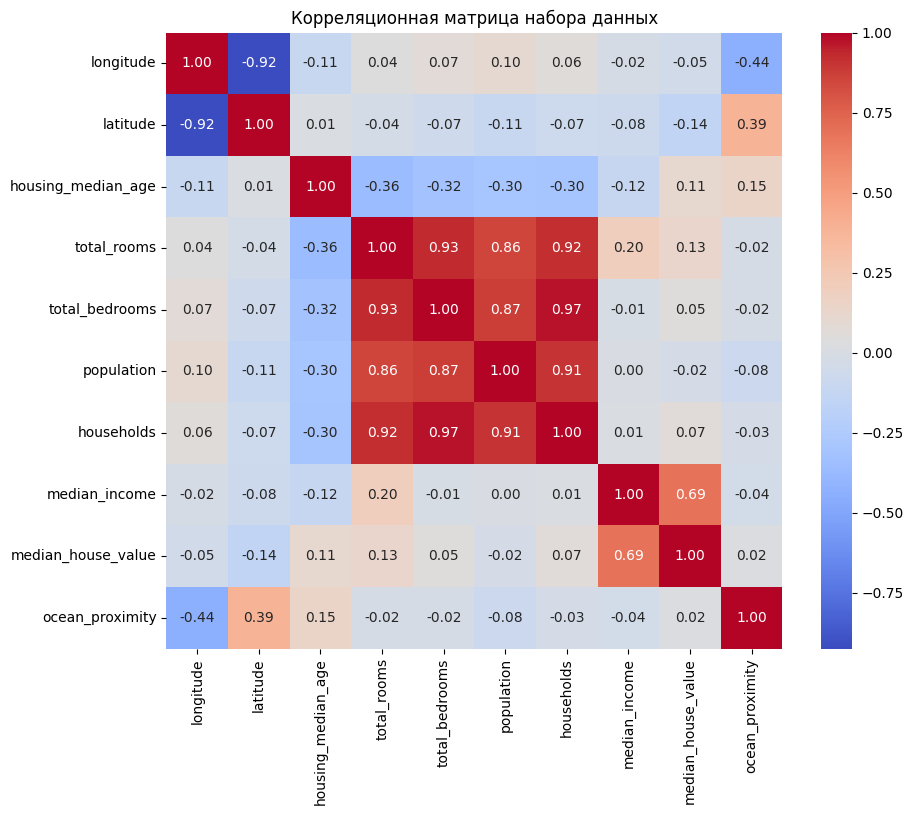
Можно наблюдать сильную корреляцию между количеством жильцов и количеством комнат, между средним доходом и средей стоимостью жилья. Умеренная - между жильцами и их средним возрастом.

### Предобработка данных

# Создание словаря для соответствия категорий и чисел  
categories\_to\_numbers = {  
 '<1H OCEAN': 1,  
 'INLAND': 2,  
 'NEAR OCEAN': 3,  
 'NEAR BAY': 4,  
 'ISLAND': 5  
}  
  
# Замена категорий числами в столбце 'ocean\_proximity'  
data['ocean\_proximity'] = data['ocean\_proximity'].map(categories\_to\_numbers)  
  
# Проверка результатов  
print(data['ocean\_proximity'].head())

0 4  
1 4  
2 4  
3 4  
4 4  
Name: ocean\_proximity, dtype: int64

# Вычисление корреляционной матрицы для числовых столбцов  
corr\_matrix = data.corr()  
  
# Создание тепловой карты корреляционной матрицы  
plt.figure(figsize=(10, 8))  
sns.heatmap(corr\_matrix, annot=True, fmt=".2f", cmap='coolwarm', square=True)  
plt.title('Корреляционная матрица набора данных')  
plt.show()



data.dtypes

longitude float64  
latitude float64  
housing\_median\_age float64  
total\_rooms float64  
total\_bedrooms float64  
population float64  
households float64  
median\_income float64  
median\_house\_value float64  
ocean\_proximity int64  
dtype: object

### Выбор метрик для оценки качества моделей

Для оценки качества моделей регрессии мы выберем следующие три метрики:

1. ***R^2*** (коэффициент детерминации): Он показывает, какую часть дисперсии зависимой переменной объясняют независимые переменные. Это очень популярная метрика для оценки качества регрессионных моделей.
2. ***MAE*** (средняя абсолютная ошибка): Данная метрика позволяет оценить среднюю абсолютную ошибку в предсказаниях, что помогает нам понять среднюю разницу между фактическим и предсказанным значениями.
3. ***RMSE*** (квадратичная средняя ошибка): Является более чувствительной к большим отклонениям между фактическим и предсказанным значениями, чем MAE, и часто используется в задачах регрессии.

# Определим функции для вычисления выбранных метрик  
def evaluate\_model(y\_true, y\_pred):  
 r2 = r2\_score(y\_true, y\_pred)  
 mae = mean\_absolute\_error(y\_true, y\_pred)  
 rmse = np.sqrt(mean\_squared\_error(y\_true, y\_pred))  
 return {"R^2": r2, "MAE": mae, "RMSE": rmse}

### Выбор моделей для решения задачи регрессии

1. Ансамблевая модель случайного леса
2. Метод К-ближайших соседей
3. Метод опорных векторов
4. Градиентный бустинг
5. Решающее дерево

models = {  
 "Random Forest": RandomForestRegressor(),  
 "K-Neighbors": KNeighborsRegressor(),  
 "Support Vector Machine": SVR(),  
 "Gradient Boosting": GradientBoostingRegressor(),  
 "Decision Tree": DecisionTreeRegressor(),   
}

### Формирование обучающей и тестовой выборок на основе исходного набора данных

data.head()

longitude latitude housing\_median\_age total\_rooms total\_bedrooms \  
0 -122.23 37.88 41.0 880.0 129.0   
1 -122.22 37.86 21.0 7099.0 1106.0   
2 -122.24 37.85 52.0 1467.0 190.0   
3 -122.25 37.85 52.0 1274.0 235.0   
4 -122.25 37.85 52.0 1627.0 280.0   
  
 population households median\_income median\_house\_value ocean\_proximity   
0 322.0 126.0 8.3252 452600.0 4   
1 2401.0 1138.0 8.3014 358500.0 4   
2 496.0 177.0 7.2574 352100.0 4   
3 558.0 219.0 5.6431 341300.0 4   
4 565.0 259.0 3.8462 342200.0 4

# Разделим наш датасет на признаки и целевую переменную  
data.dropna()  
x = data.drop(columns=['median\_house\_value'])  
y = data['median\_house\_value']  
  
# Разделим данные на обучающую и тестовую выборки  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=42)

### Построим базовые модели и сделаем предсказания

base\_results = {}  
imputer = SimpleImputer(strategy='most\_frequent')  
x\_train = imputer.fit\_transform(x\_train)  
x\_test = imputer.transform(x\_test)  
  
  
for name, model in models.items():  
 model.fit(x\_train, y\_train)  
 y\_pred = model.predict(x\_test)  
 base\_results[name] = evaluate\_model(y\_test, y\_pred)  
  
print("Baseline Results")  
for name, metrics in base\_results.items():  
 print(f"{name}: R^2: {metrics['R^2']:.4f}, MAE: {metrics['MAE']:.4f}, RMSE: {metrics['RMSE']:.4f}")

Baseline Results  
Random Forest: R^2: 0.8104, MAE: 32009.1166, RMSE: 49844.2873  
K-Neighbors: R^2: 0.2237, MAE: 78365.3029, RMSE: 100856.5830  
Support Vector Machine: R^2: -0.0487, MAE: 87342.2458, RMSE: 117224.9936  
Gradient Boosting: R^2: 0.7597, MAE: 38634.9416, RMSE: 56116.8710  
Decision Tree: R^2: 0.6506, MAE: 43323.6873, RMSE: 67668.4107

### Подберем гиперпараметры выбранных моделей

# Определим гиперпараметры для каждой модели  
# Случайный лес  
rf\_params = {  
 'n\_estimators': [100, 200],  
 'max\_depth': [None, 10, 20],  
 'min\_samples\_split': [2, 5],  
 'min\_samples\_leaf': [1, 2]  
}  
  
# K-Neighbors Regressor  
knn\_params = {  
 'n\_neighbors': [3, 5, 10],  
 'weights': ['uniform', 'distance'],  
 'algorithm': ['auto', 'ball\_tree', 'kd\_tree', 'brute']  
}  
  
# Метод опорных векторов  
svr\_params = {  
 'kernel': ['linear', 'rbf'],  
 'C': [0.1, 1, 10],  
 'gamma': [0.01, 0.1, 1]  
}  
  
# Градиентный бустинг  
gb\_params = {  
 'n\_estimators': [100, 200],  
 'learning\_rate': [0.01, 0.1, 0.2],  
 'subsample': [0.7, 1.0],  
 'max\_depth': [3, 7, 10]  
}  
  
# Решающее дерево  
dt\_params = {  
 'max\_depth': [None, 10, 20],  
 'min\_samples\_split': [2, 5, 10],  
 'min\_samples\_leaf': [1, 2, 4]  
}  
  
params = {  
 "Random Forest": rf\_params,  
 "K-Neighbors": knn\_params,  
 "Support Vector Machine": svr\_params,  
 "Gradient Boosting": gb\_params,  
 "Decision Tree": dt\_params,   
}  
  
# Используем GridSearchCV для подбора оптимальных параметров для каждой модели  
best\_params = {}  
for name, model in models.items():  
 print(f"Подбор гиперпараметров для {name}...")  
 grid\_search = GridSearchCV(model, params[name], cv=3, scoring='neg\_mean\_absolute\_error')  
 grid\_search.fit(x\_train, y\_train)  
 best\_params[name] = grid\_search.best\_params\_  
  
print('Лучшие гиперпараметры для каждой модели:')  
for name, params in best\_params.items():  
 print(f"{name}: {params}")

Подбор гиперпараметров для Random Forest...  
Подбор гиперпараметров для K-Neighbors...  
Подбор гиперпараметров для Support Vector Machine...  
Подбор гиперпараметров для Gradient Boosting...  
Подбор гиперпараметров для Decision Tree...  
Лучшие гиперпараметры для каждой модели:  
Random Forest: {'max\_depth': None, 'min\_samples\_leaf': 2, 'min\_samples\_split': 5, 'n\_estimators': 200}  
K-Neighbors: {'algorithm': 'auto', 'n\_neighbors': 10, 'weights': 'distance'}  
Support Vector Machine: {'C': 10, 'gamma': 0.01, 'kernel': 'linear'}  
Gradient Boosting: {'learning\_rate': 0.1, 'max\_depth': 10, 'n\_estimators': 200, 'subsample': 0.7}  
Decision Tree: {'max\_depth': 10, 'min\_samples\_leaf': 4, 'min\_samples\_split': 10}

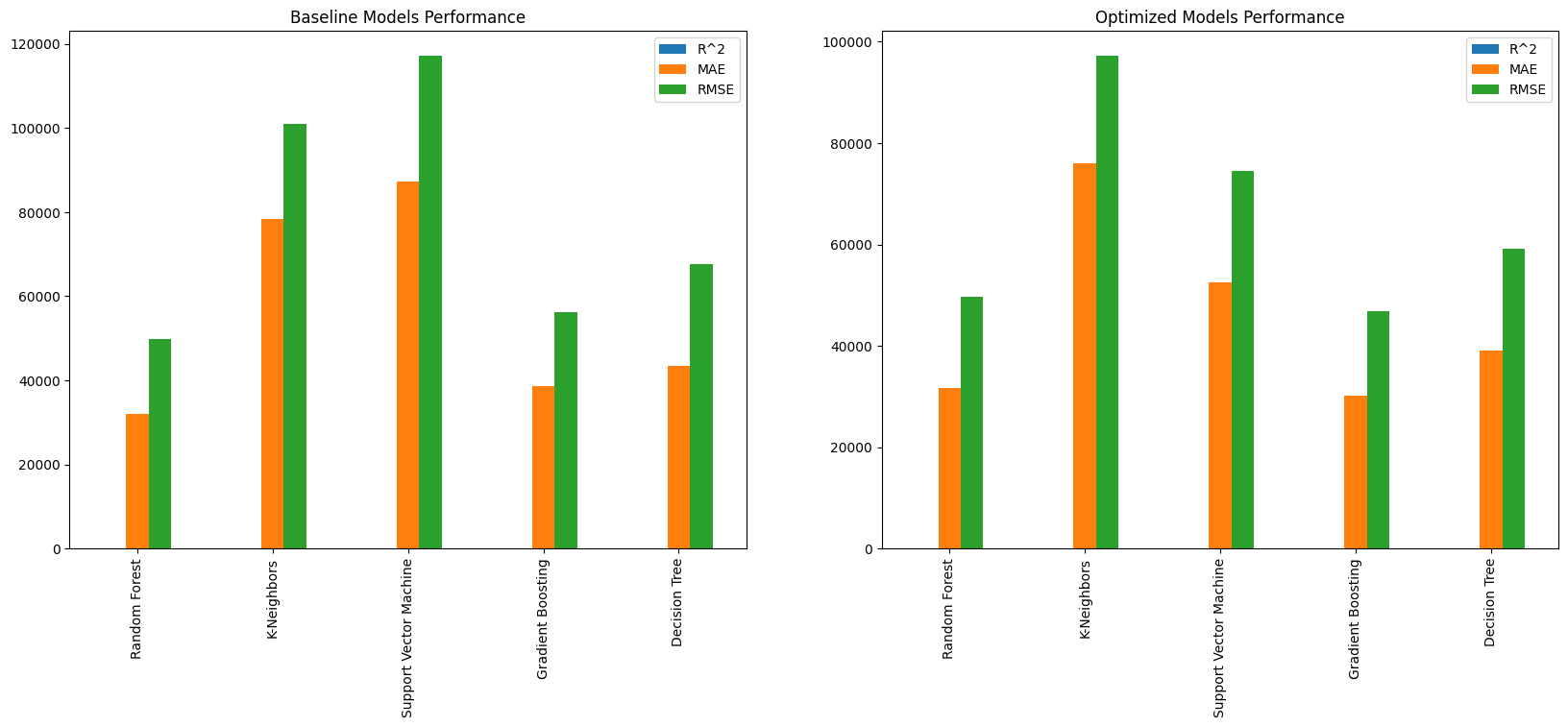
# Создаем новые модели с лучшими гиперпараметрами  
optimized\_models = {  
 "Random Forest": RandomForestRegressor(\*\*best\_params["Random Forest"]),  
 "K-Neighbors": KNeighborsRegressor(\*\*best\_params["K-Neighbors"]),  
 "Support Vector Machine": SVR(\*\*best\_params["Support Vector Machine"]),  
 "Gradient Boosting": GradientBoostingRegressor(\*\*best\_params["Gradient Boosting"]),  
 "Decision Tree": DecisionTreeRegressor(\*\*best\_params["Decision Tree"]),   
}  
  
# Обучение и оценка моделей с подобранными гиперпараметрами  
optimized\_results = {}  
for name, model in optimized\_models.items():  
 model.fit(x\_train, y\_train)  
 y\_pred = model.predict(x\_test)  
 optimized\_results[name] = evaluate\_model(y\_test, y\_pred)  
  
print("Результаты моделей с подобранными гиперпараметрами")  
for name, metrics in optimized\_results.items():  
 print(f"{name}: R^2: {metrics['R^2']:.4f}, MAE: {metrics['MAE']:.4f}, RMSE: {metrics['RMSE']:.4f}")

Результаты моделей с подобранными гиперпараметрами  
Random Forest: R^2: 0.8118, MAE: 31760.6819, RMSE: 49657.8142  
K-Neighbors: R^2: 0.2779, MAE: 75993.6938, RMSE: 97278.4378  
Support Vector Machine: R^2: 0.5754, MAE: 52560.9242, RMSE: 74591.7022  
Gradient Boosting: R^2: 0.8333, MAE: 30171.5567, RMSE: 46742.1416  
Decision Tree: R^2: 0.7332, MAE: 39160.4298, RMSE: 59125.3753

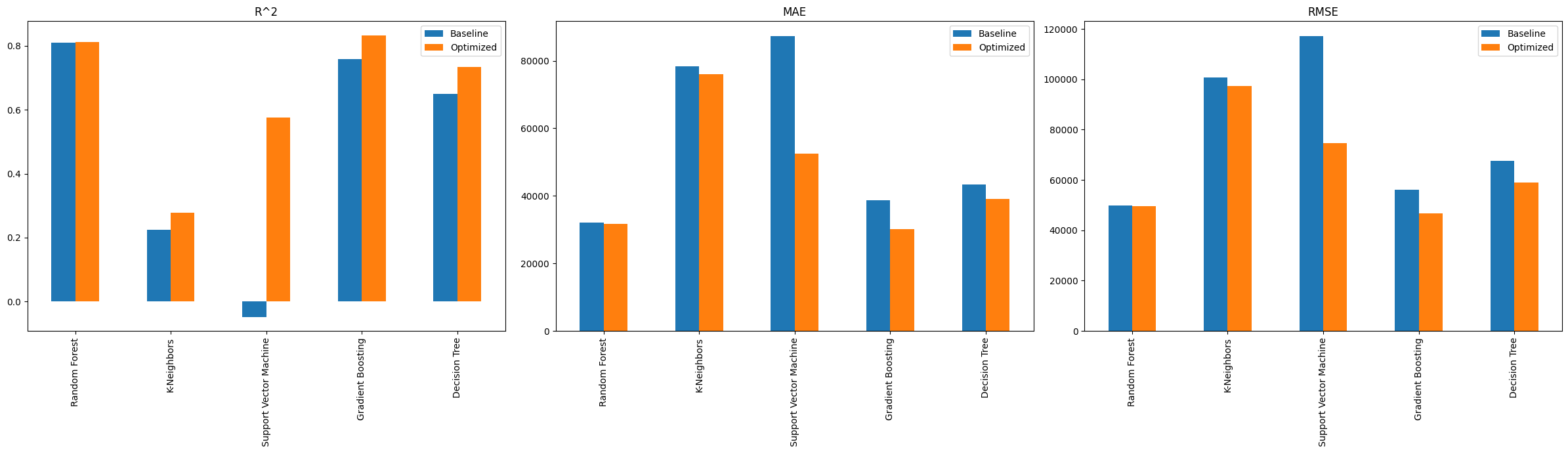
### Сравним базовые и настроенные модели

# Компилируем результаты базовых моделей и оптимизированных моделей  
comparison\_results = {"Базовые": base\_results, "Оптимизированные": optimized\_results}  
  
for category, results in comparison\_results.items():  
 print(f"\n{category} Модели:")  
 for name, metrics in results.items():  
 print(f"{name}: R^2: {metrics['R^2']:.4f}, MAE: {metrics['MAE']:.4f}, RMSE: {metrics['RMSE']:.4f}")  
  
# Визуализируем результаты базовых моделей и оптимизированных моделей  
baseline\_df = pd.DataFrame(base\_results).T  
optimized\_df = pd.DataFrame(optimized\_results).T  
  
fig, axes = plt.subplots(1, 2, figsize=(20, 7))  
  
baseline\_df.plot(kind='bar', ax=axes[0], title='Baseline Models Performance')  
optimized\_df.plot(kind='bar', ax=axes[1], title='Optimized Models Performance')  
print('\n')  
plt.show()  
print('\n')

Базовые Модели:  
Random Forest: R^2: 0.8104, MAE: 32009.1166, RMSE: 49844.2873  
K-Neighbors: R^2: 0.2237, MAE: 78365.3029, RMSE: 100856.5830  
Support Vector Machine: R^2: -0.0487, MAE: 87342.2458, RMSE: 117224.9936  
Gradient Boosting: R^2: 0.7597, MAE: 38634.9416, RMSE: 56116.8710  
Decision Tree: R^2: 0.6506, MAE: 43323.6873, RMSE: 67668.4107  
  
Оптимизированные Модели:  
Random Forest: R^2: 0.8118, MAE: 31760.6819, RMSE: 49657.8142  
K-Neighbors: R^2: 0.2779, MAE: 75993.6938, RMSE: 97278.4378  
Support Vector Machine: R^2: 0.5754, MAE: 52560.9242, RMSE: 74591.7022  
Gradient Boosting: R^2: 0.8333, MAE: 30171.5567, RMSE: 46742.1416  
Decision Tree: R^2: 0.7332, MAE: 39160.4298, RMSE: 59125.3753



# Превращаем результаты в DataFrame  
baseline\_df = pd.DataFrame(base\_results).T  
optimized\_df = pd.DataFrame(optimized\_results).T  
  
# Создаем графики для каждой метрики  
metrics = ['R^2', 'MAE', 'RMSE']  
fig, axes = plt.subplots(1, 3, figsize=(24, 7))  
  
for i, metric in enumerate(metrics):  
 metric\_data = pd.DataFrame({  
 'Baseline': baseline\_df[metric],  
 'Optimized': optimized\_df[metric]  
 })  
 metric\_data.plot(kind='bar', ax=axes[i], title=metric)  
  
plt.tight\_layout()  
print('\n')  
plt.show()  
print('\n')



## Заключение и выводы

#### Описание исследования

В рамках исследования был проведен анализ эффективности разнообразных алгоритмов машинного обучения применительно к задачам регрессионного анализа. В числе протестированных моделей оказались дерево решений, метод векторов опоры, ансамбль случайных лесов, техника градиентного бустинга и алгоритм K-ближайших соседей.

#### Сравнение моделей

Для каждой модели были измерены три метрики: ( R^2 ) (коэффициент детерминации), MAE (средняя абсолютная ошибка) и RMSE (корень из средней квадратической ошибки). Сравнение проводилось в двух этапах: до оптимизации гиперпараметров (Базовые Модели) и после их оптимизации (Оптимизированные Модели).

**Базовые Модели:**

* **Random Forest: R^2:** 0.8104, MAE: 32009.1166, RMSE: 49844.2873
* **K-Neighbors: R^2:** 0.2237, MAE: 78365.3029, RMSE: 100856.5830
* **Support Vector Machine:** R^2: -0.0487, MAE: 87342.2458, RMSE: 117224.9936
* **Gradient Boosting: R^2:** 0.7597, MAE: 38634.9416, RMSE: 56116.8710
* **Decision Tree: R^2:** 0.6506, MAE: 43323.6873, RMSE: 67668.4107

**Оптимизированные Модели:**

* **Random Forest: R^2:** 0.8118, MAE: 31760.6819, RMSE: 49657.8142
* **K-Neighbors: R^2:** 0.2779, MAE: 75993.6938, RMSE: 97278.4378
* **Support Vector Machine:** R^2: 0.5754, MAE: 52560.9242, RMSE: 74591.7022
* **Gradient Boosting:** R^2: 0.8333, MAE: 30171.5567, RMSE: 46742.1416
* **Decision Tree: R^2:** 0.7332, MAE: 39160.4298, RMSE: 59125.3753

## Итоги

1. **Повышение эффективности через настройку гиперпараметров:** Оптимизация гиперпараметров привела к улучшению всех рассмотренных моделей. Это подчеркивает значимость тонкой настройки гиперпараметров для повышения производительности алгоритмов. К примеру, коэффициент детерминации ( R^2 ) для модели дерева решений возрос с 0.6506 до 0.7332, а средняя абсолютная ошибка (MAE) сократилась с 43323.6873 до 39160.4298.
2. **Наивысшая эффективность:**
   * Самые точные результаты среди оптимизированных алгоритмов продемонстрировала модель случайного леса, достигнув ( R^2 = 0.8118 ), с MAE = 31760.6819 и RMSE = 49657.8142, что говорит о её превосходстве в точности предсказаний.
   * Модель градиентного бустинга также показала высокие показатели (( R^2 = 0.8333 ), MAE = 30171.5567, RMSE = 46742.1416), слегка уступая модели случайного леса по ключевым метрикам.
3. **Минимальная эффективность:**
   * Метод опорных векторов продемонстрировал наименее удовлетворительные результаты до и после настройки гиперпараметров. Это особенно заметно по начальным данным, где ( R^2 = -0.0487 ), указывая на неспособность модели предсказывать результаты лучше, чем базовое среднее.
4. **Результаты других алгоритмов:**
   * Дерево решений и K-ближайших соседей также улучшили свои показатели после корректировки гиперпараметров, однако они не достигли уровня случайного леса и градиентного бустинга.

## Вывод

Исходя из проведённого анализа, для регрессионных задач на выбранном наборе данных предпочтительнее использовать модель случайного леса или градиентного бустинга, предварительно проведя детальную настройку гиперпараметров. Эти алгоритмы показали лучшие результаты в сравнении с остальными и способны обеспечить точные и стабильные прогнозы.

**Список литературы:**

1. Линейная регрессия (Linear regression) – Текст : электронный // Loginom : [сайт]. – 2021. – URL: https://wiki.loginom.ru/articles/linear-regression.html (дата обращения : 27.03.2021). – Режим доступа : свободный.

2. Переобучение. – Текст : электронный // Wikipedia: свободная энциклопедия : [сайт]. – URL: https://ru.wikipedia.org/wiki/Переобучение (дата обращения : 27.03.2021). – Режим доступа : свободный.

3. An Introduction to Statistical Learning: with Applications in R / G. James, D. Witten, T. Hastie, R. Tibshirani. – New York : Springer, 2013. – 426 p. – Text : direct.

4. McDonald, John H. Multiple regression. – Text : electronic // Handbook of Biological Statistics : [website]. – 2014. – URL: http://www.biostathandbook.com/multipleregression.html (date of treatment : 27.03.2021). – Access mode: free. 5. Strang Gilbert Differential Equations and Linear Algebra Wellesley. – Cambridge, 2014. – 512 p. – Text : direct.