Московский авиационный институт

(национальный исследовательский университет)

«МАИ»

Факультет №3 — «Системы управления, информатика и электроэнергетика»

Кафедра 316 — «Системное моделирование и автоматизированное проектирование»

Курсовая работа

по дисциплине: **«Машинное обучение на больших данных»**

на тему: **«Сравнение алгоритмов регрессионного анализа»**

Выполнил:

студент группы М3О-118М-19

Пономарев Роман

Москва 2020

Задача регресии в машинном обучении

Алгоритмы машинного обучения можно описать как обучение целевой функции f, которая наилучшим образом соотносит входные переменные X и выходную переменную Y: Y = f(X).

Мы не знаем, что из себя представляет функция f. Ведь если бы знали, то использовали бы её напрямую, а не пытались обучить с помощью различных алгоритмов.

Наиболее распространённой задачей в машинном обучении является предсказание значений Y для новых значений X. Это называется прогностическим моделированием, и наша цель — сделать как можно более точное предсказание.

Целью данной работы является прогнозирование, применение известных методов регрессии на определенной выборке данных. Применив различные алгоритмы на одних и тех же данных, можно достаточно точно провести сравнение данных методов по соответствию их прогнозов к заранее известной выборке данных.

Исходные данные

В качестве данных, нужных для тестирования методов машинного обучения взят открытый для свободного доступа датасет с сайта https://www.kaggle.com/vitaliymalcev/russian-passenger-air-service-20072020/data?select=russian_passenger_air_service_2.csv

Данные содержат информацию о пассажиропотоке аэропортов РФ за временной период от 2007 до 2020 годов.

Исходя из ситуации, сложившейся в 2020 году, пассажиропоток за этот год существенно снизился, близок к нулю, что делает этот год “выбросом” в выборке. Поэтому в данной работе, данные за 2020 год не рассматриваются.

Также, ввиду того, что в рамках данной работы реализуются алгоритмы простой линейной регрессии и бутстрепа выборкой для исследования являются небольшие данные по среднему пассажиропотоку суммарно всех аэропортов за год в периоде от 2007 до 2020 года. Так, именно этот признак является ключевым параметром, подлежащим прогнозированию и последующему сравнению между алгоритмами.

Для дальнейшего исследования, доступные данные делятся пополам – на тренировочную и тестовую выборки.

Представление исходных данных:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
# import seaborn as sns
# %matplotlib inline
# %config InlineBackend.figure_format = 'svg'
df = pd.read_csv('russian_passenger_air_service_2.csv', sep=',')
print("'ерхние пять строчек исходного датасета: ")
print(df.head())
#Отбросим данные за 2020 год(из-за коронавируса)
df = df.drop(np.where(df['Year'] == 2020)[0])

#Отбросим колонку с координатами аэропрота
df.drop(["Airport coordinates"], axis = 1, inplace = True)

#Найдем средний пассажиропоток аэропортов по годам(кроме 2020)
data = df.groupby("Year").mean()
print("змененный датасет под исследование:")
print(data)

mean_years = data["Whole year"].tolist()
print("Средние значения по годам", mean_years)

years = data.index.tolist()
print("Список годов = ", years)


plt.title("Средний пассажиропоток по годам(исходные данные)")
plt.xlabel("""од")
plt.ylabel("Пассажиропоток(человек)")
plt.plot(years, mean_years)
plt.show()

#Построение гистограммы:
#x1 = range(len(mean_years))
#x1 = range(years)

ax = plt.gca()
#ax.bar(x1, mean_years, align='edge') #align='edge' - выравнивание по
границе, а не по центр
#ax.set_xticks(x1)
ax.bar(years, mean_years)
ax.set_title('Средний пассажиропоток по годам(исходные данные) - "истограмм
a')
ax.set_xlabel('"од')
ax.set_ylabel('Пассажиропоток(человек)')

#ax.set_xticklabels(('first', 'second', 'third', 'fourth'))
plt.show()
```

https://github.com/romanponomarew/Machine-learning/blob/master/kursach(ML).py

Рисунок 1 - Исходные данные(График)

Рисунок 2 - Исходные данные (Гистограмма)

Прогнозирование методом простой линейной регресии

Линейная регрессия – одна из важнейших и широко используемых техник регрессии. Эта самый простой метод регрессии. Одним из его достоинств является лёгкость интерпретации результатов.

Регрессия ищет отношения между переменными.

Для примера можно взять сотрудников какой-нибудь компании и понять, как значение зарплаты зависит от других **данных**, таких как опыт работы, уровень образования, роль, город, в котором они работают, и так далее.

Регрессия решает проблему единого **представления** данных анализа для каждого работника. Причём опыт, образование, роль и город – это независимые переменные при зависимой от них зарплате.

Таким же способом можно установить математическую зависимость между ценами домов в определённой области, количеством комнат, расстоянием от центра и т. д.

Регрессия рассматривает некоторое явление и ряд наблюдений. Каждое наблюдение имеет две и более переменных. Предполагая, что одна переменная зависит от других, вы пытаетесь построить отношения между ними.

Другими словами, **вам нужно найти функцию, которая отображает зависимость одних переменных или данных от других.**

Зависимые данные называются **зависимыми переменными, выходами** или **ответами.**

Независимые данные называются **независимыми переменными, входами или предсказателями.**

Обычно в регрессии присутствует одна непрерывная и неограниченная зависимая переменная. Входные переменные могут быть неограниченными, дискретными или категорическими данными, такими как пол, национальность, бренд, и т.д.

Регрессия полезна **для прогнозирования ответа** на новые условия. Можно угадать потребление электроэнергии в жилом доме из данных температуры, времени суток и количества жильцов.

Линейная регрессия — это **линейная модель**, которая предполагает линейную связь между **входными переменными (Xi)** и **единственной переменной на вывода (Y)**. Более конкретно, что **y** может быть рассчитана через линейную комбинации входных переменных (X) (или У = B1*X1 + B2*X2 + ..Bn*Xn).

При наличии одной переменной ввода (x) метод называется **простой линейной регрессией**. Когда существует несколько переменных входных данных, литература из статистики часто называет метод **множественной линейной регрессией.**

Различные методы могут быть использованы для подготовки или обучения линейной регрессии. Наиболее распространенным из которых называется Метод наименьших квадратов (или сокращенно МНК, по-английски это Ordinary Least Squares или OLS).

Представление - это линейное уравнение, объединяющее определенный набор входных значений (X) решений, к которому является прогнозируемый вывод для этого набора входных значений (y). Таким образом, как значения входных данных (x), так и выходное значение являются числовыми.

Линейное уравнение присваивает **масштабный коэффициент** (по-английски "scale factor") к каждому входному значению X. Масштабный коэффициент представлен греческой буквой Beta (B). Добавлен также один дополнительный коэффициент, добавляющую дополнительную степень свободы (например, движение вверх и вниз по двумерном участку) и часто называют **коэффициентом перехвата или смещения** (по-английски "bias coefficient").

Наиболее простая задача регрессии когда на вход подается одна переменная X и есть одно выходящее значение Y. Форма подобной модели будет:

Y = B0 + B1 * X

В случае многомерных измерений (т.е. когда у нас есть более одной вводной переменной (X)), линия превращается в плоскостью или гипер-плоскости. Таким образом, представление представляет собой форму уравнения и конкретные значения, используемые для коэффициентов (например, B0 и B1 в приведенном выше примере).

Когда говорят о сложности регрессионной модели, такой как линейная регрессия - имеют ввиду количество коэффициентов, используемых в модели.

Когда конкретный элемент коэффициент Beta становится нулевым, он эффективно удаляет влияние входной переменной на модель и, следовательно, влияния на прогноз модели (0 * Xi = 0). Это становится актуальным, если вы применяете методы регуляризации (о них мы расскажем отдельно), которые изменяют алгоритм обучения, чтобы уменьшить сложность моделей регрессии, оказывая давление на абсолютный размер коэффициентов, приводя некоторые из них к нулю.

При простой линейной регрессии, когда у нас есть один входной параметр, мы можем использовать статистику для оценки коэффициентов.

Для этого необходимо вычислить статистические свойства на таких данных, как среднее значение, стандартные отклонения, корреляции и ковариантность. Все данные должны быть доступны для обхода и расчета статистик.

Код алгоритма:

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
# import seaborn as sns
# %matplotlib inline
# %config InlineBackend.figure_format = 'svg'
df = pd.read_csv('russian_passenger_air_service_2.csv', sep=',')

#Отбросим данные за 2020 год(из-за коронавируса)
df = df.drop(np.where(df['Year'] == 2020)[0])
print(df.head())

#Отбросим колонку с координатами аэропрота
df.drop(["Airport coordinates"], axis = 1, inplace = True)
print(df.head())

#Найдем средний пассажиропоток аэропортов по годам(кроме 2020)
data = df.groupby("Year").mean()
print(data)

mean_years = data["Whole year"].tolist()
#mean_years = data["Whole year"]
print("Средние значения по годам", mean_years)

#years = data.index.tolist()
years = data.index.tolist()
print("Список годов = ", years)


# регрессия с использованием набора данных
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
import pandas as pd

y = np.array(mean_years)
```

```
x = np.array(years).reshape((-1, 1))


# Разделяем данные на обучающие / тестовые наборы
#Первая половина доступной выборки - обучающая
x_train = x[:(len(x)//2)]
#�'торая половина доступной выборки - тестовая, для проверки
x_test = x[(len(x)//2):]
print("x_train=", x_train)
print("x_test=", x_test)

#Такой же принцип и с Y - разделение на обучающие и тестовые наборы
y_train = y[:(len(y)//2)]
y_test = y[(len(y)//2):]
#print("y_train=", y_train)
#print("y_test=", y_test)

# Сюжетные выходы
plt.scatter(x_test, y_test,  color='black')

plt.title('Test Data')
plt.xlabel('Size')

plt.ylabel('Price')
plt.xticks(())
plt.yticks(())

#Создаем экземпляр класса LinearRegression, это будет моделью линейной
регрессии
model = linear_model.LinearRegression()

# Тренируем модель, используя тренировочные наборы(автоматический подбор
параметров модели)
model.fit(x_train, y_train)

# Сюжетные выходы
#Красным - прогнозируемый результат методом линейной регрессии
plt.plot(x_test, model.predict(x_test), color='red', linewidth=3)
#Синим - настоящие результаты в тестовой выборке
plt.plot(x_test, y_test, color='blue', linewidth=3)
#�-еленым - тренировочная выборка
plt.plot(x_train, y_train, color='green', linewidth=3)
plt.show()
#На этом графике, мы наносим тестовые данные. Красная линия указывает линию
наилучшего соответствия для прогнозирования Y.

#�ндивидуальный прогноз с использованием модели линейной регрессии:

y_pred = model.predict(x_test)
print(f'Прогнозируемый пассажиропоток за годы от {x_test[0]} до
{x_test[len(x_test)-1]} равен', y_pred, sep='\n')
print(f"Реальный пассажиропоток за годы {x_test[0]} до {x_test[len(x_test)-
1]} равен ", y_test)
```

https://github.com/romanponomarew/Machine-learning/blob/master/regression(ML).py

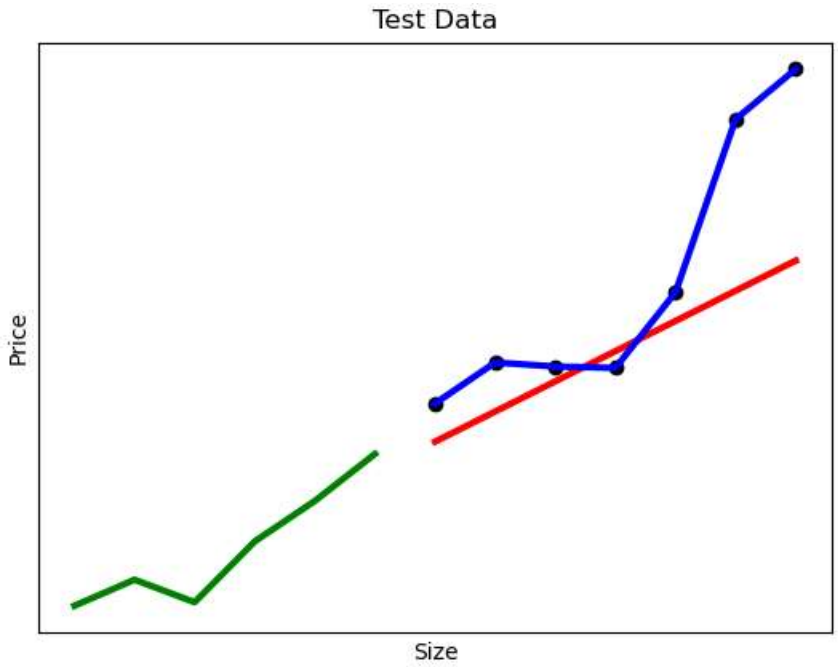

Рисунок 3 - Линейная регрессия

Таким образом, изучив рисунок 3(зеленый цвет – тренировочная выборка, синий цвет – тестовая выборка, красный цвет – спрогнозированная линейная зависимость), можно прийти к выводу, что метод простой линейной регрессии позволяет получить данные, сильно отклоняющиеся от реальных. Причиной этому служит то, что исходные данные слабо подвержены какой-либо однозначной зависимости, несоответствия их линейной зависимости, а также тот факт, что в тренировочных и тестовых данных присутствуют выбросы.

Поэтому можно сделать вывод, что хоть алгоритм простой линейной регрессии и является простым и удобным, годится он лишь для выявления общей тенденции изменения данных – возрастания либо убывания и примерную степень изменчивости данных. Прогнозирование же конкретных данных этим методом приводит к недопустимым погрешностям.

Прогнозирование бутстреп методом

Задача — спрогнозировать дальнейшее развитие параметра, описанного в исходной выборке временного ряда. Прогноз необходимо реализовать, используя метод бутстрепа для размножения исходной выборки.

Дан средний пассажиропоток за год , всего рассмотрено 12 лет, поэтому из ряда мы случайным образом возьмем 12 значений и сделаем это 1 000 раз. Далее по каждому из 1 000 рядов со случайными значениями в выборке рассчитаем среднее значение каждого ряда, и по средним значениям каждого ряда рассчитаем еще раз среднее, получим средний годовой пассажиропоток.

Методы размножения выборок (бутстреп-методы)

Бутстрэп (англ. *bootstrap*) в статистике — практический компьютерный метод исследования распределения статистик вероятностных распределений, основанный на многократной генерации выборок методом Монте-Карло на базе имеющейся выборки[2]. Позволяет просто и быстро оценивать самые разные статистики (доверительные интервалы, дисперсию, корреляцию и так далее) для сложных моделей.

Эконометрика и прикладная статистика бурно развиваются последние десятилетия. Серьезным (хотя, разумеется, не единственным и не главным) стимулом является стремительно растущая производительность вычислительных средств. Поэтому понятен острый интерес к статистическим методам, интенсивно использующим компьютеры. Одним из таких методов является так называемый "бутстреп", предложенный в 1977 г. Б.Эфроном из Станфордского университета (США).

Сам термин "бутстреп" - это "bootstrap" русскими буквами и буквально означает что-то вроде: "вытягивание себя (из болота) за шнурки от ботинок". Термин специально придуман и заставляет вспомнить о подвигах барона Мюнхгаузена.

Метод бутстрэпа заключается в следующем. Пусть имеется выборка **X** размера **N**. Равномерно возьмем из выборки **N** объектов с возвращением. Это означает, что мы будем **N** раз выбирать произвольный объект выборки (считаем, что каждый объект «достается» с одинаковой вероятностью **1N**), причем каждый раз мы выбираем из всех исходных **N** объектов. Можно представить себе мешок, из которого достают шарики: выбранный на каком-то шаге шарик возвращается обратно в мешок, и следующий выбор опять делается равновероятно из того же числа шариков.

Отметим, что из-за возвращения среди них окажутся повторы. Обозначим новую выборку через **X1**. Повторяя процедуру **M** раз, сгенерируем **M** подвыборок **X1,…,XM**. Теперь мы имеем достаточно большое число выборок и можем оценивать различные статистики исходного распределения.

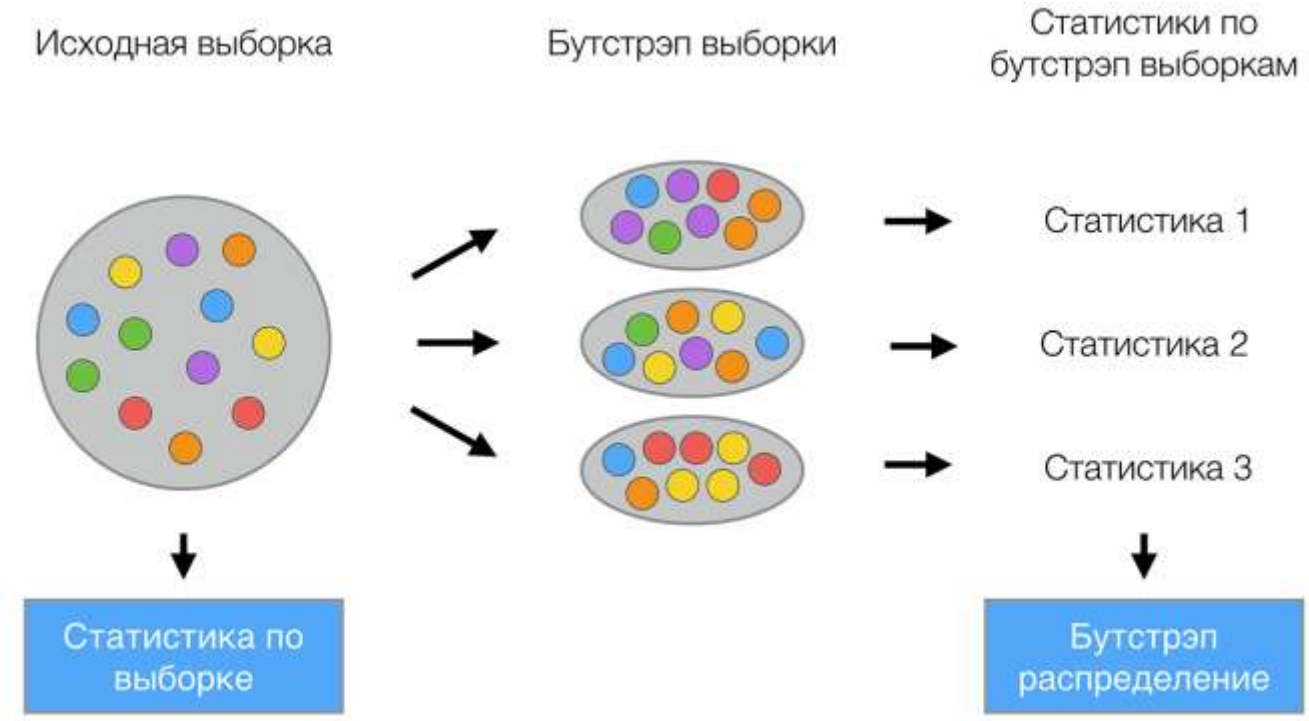

Рисунок 4 - Бутстреп метод

Алгоритм решения задачи:

1. Аппроксимация исходных данных, применяя метод наименьших квадратов, с целью нахождения уравнения прямой и ее построения. Отобразить прямую графически для наглядности.
2. Создать список из значений расстояний **h** между каждой из данных точек и получившейся прямой.
3. Применяя метод бутстрепа увеличить список расстояний до большего значения (например, 1000 значений). Создать, таким образом, несколько списков большего размера (например, 10 списков).
4. Найти среднее значение для каждого из получившихся списков и занести значения в новый список.
5. Найти максимальное и минимальное значения в получившемся списке средних значений. Интервал между максимальным и минимальным

значением – доверительный интервал для расстояний от точки до прямой, согласно методу бутстрепа.

6. Теперь, используя полученные значения, можно сделать прогноз на значение Y (цена), подставив в уравнение значение X(месяц). В итоге, получаем прогноз для Y, значение которого лежит в небольшом интервале.

Код Bootstrap forecast:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
# import seaborn as sns
# %matplotlib inline
# %config InlineBackend.figure_format = 'svg'
df = pd.read_csv('russian_passenger_air_service_2.csv', sep=',')

#Отбросим данные за 2020 год(из-за коронавируса)
df = df.drop(np.where(df['Year'] == 2020)[0])
print(df.head())

#Отбросим колонку с координатами аэропрота
df.drop(["Airport coordinates"], axis = 1, inplace = True)
print(df.head())

#Найдем средний пассажиропоток аэропортов по годам(кроме 2020)
data = df.groupby("Year").mean()
print(data)

mean_years = data["Whole year"].tolist()
print("Средние значения по годам", mean_years)

years = data.index.tolist()
print("Список годов = ", years)



#МНК - Метод наименьших квадратов
#У нас есть множество точек(y = среднее количество пассажиров за год, x =
год). Через них надо провести прямую, которая как можно ближе проходила к
этим точкам.

#Проведем прямую y = kx + b через данные точки

x = np.array(years)
y = np.array(mean_years)

# Разделяем данные на обучающие / тестовые наборы
#Первая половина доступной выборки - обучающая
x_train = x[:(len(x)//2)]
#'торая половина доступной выборки - тестовая, для проверки
x_test = x[(len(x)//2):]
print("x_train=", x_train)
print("x_test=", x_test)

#Такой же принцип и с Y - разделение на обучающие и тестовые наборы
y_train = y[:(len(y)//2)]
```

```
y_test = y[(len(y)//2):]

data1 = dict(zip(x_train, y_train))
print(data1)

#Перепишем линейное уравнение y = mx + c как y = Rp, где A = [[ x 1 ]] и p =
[[m], [c]]
#Построим R по x :
R = np.vstack([x_train, np.ones(len(x_train))]).T
#спользуем lstsq для решения его относительно вектора p
k, b = np.linalg.lstsq(R, y_train)[0]
#print(k, b)

def show_graf(x, y, k, b):
    #Построим график полученной прямой и укажем на нем точки
    plt.plot(x, y, 'o', label='Original data', markersize=10)
    plt.plot(x, k*x + b, 'r', label='Fitted line')
    plt.legend()
    plt.show()
#show_graf(x, y, k, b)

#Рассчитаем кратчайшие расстояния от исходных данных(точек) до полученной
прямой
h = abs(k*x_train-1*y_train+b)/((k**2 + (-1)**2)**0.5)
#Рассчитаем кратчайшие расстояния от исходных данных(точек) до полученной
прямой
h = abs(k*x_train-1*y_train+b)/((k**2 + (-1)**2)**0.5)
print("Список полученных расстояний:", "\n", h)

#Функция для отображения гистограммы для передаваемого списка(h, y)
def show_hist(h):
    x1 = range(len(h))
    ax = plt.gca()
    ax.bar(x1, h, align='edge') # align='edge' - выравнивание по границе, а
не по центру
    ax.set_xticks(x1)
    #ax.set_xticklabels(('first', 'second', 'third', 'fourth'))
    plt.show()

#дея применения бутсрэпа в том, что у нас есть выборка небольшого размера и
нам надо оценить, например, среднее.
# 'место подсчета среднего самой этой выборки, мы извлекаем n_samples
выборок с возвращением (то есть элементы могут повторяться) из исходной.
# У полученных выборок считаем среднее. Его уже оцениваем, вместо оценки
среднего исходной выборки.
def get_bootstrap_samples(data, n_samples):
    indices = np.random.randint(0, len(data), (n_samples, len(data)))
    samples = data[indices]
    return samples
#Получим новую выборку из 1000 значений:
n_samples = 1000
number = 10
#Функция расчета доверительного интервала бутстреп методом(прогноз расстояния
h):
def bootstrap_forecast(h, number, n_samples):

    spisok =[]
    for i in range(0, number):
        sample = get_bootstrap_samples(h, n_samples)
        spisok1 = spisok.append(np.mean(sample))
    #print("spisok =", spisok)
    global h_min
    global h_max
```

```
    h_min = min(spisok)
    h_max = max(spisok)
    print("Минимум доверительного интервала для расстояния =", h_min)
    print("Максимум доверительного интервала для расстояния =", h_max)

#bootstrap_forecast(h, number, n_samples)
#Прогноз будущих точек:
# Так, например, двигаясь по оси Ox, значение У будет принадлжать интервалу:
#h = abs(k*x-1*y+b)/((k**2 + (-1)**2)**0.5)
#Функция для расчета пассажиропотока на определенный месяц:

def forecast(x, h_min, h_max, y_pred):

    #�'ыше прямой МНК:
    y_max1 = -h_min*((k**2 + (-1)**2)**0.5) + k*x + b
    y_min1 = -h_max*((k**2 + (-1)**2)**0.5) + k*x + b
    #Ниже прямой МНК:
    y_min2 = h_min*((k**2 + (-1)**2)**0.5) - k*x - b
    y_max2 = h_max*((k**2 + (-1)**2)**0.5) - k*x - b
    #y = h*((k**2 + (-1)**2)**0.5) - k*x - b
    print(f"Максимальный прогнозируемый пассажиропоток для {x} года =
",y_max1)
    print(f"Минимальный прогнозируемый пассажиропоток для {x} года =
",y_min1)
    passengers = (y_max1 + y_min1)/2
    print(f"Прогнозируемый пассажиропоток за {x} год равен", passengers)
    y_pred.append(passengers)
    #print("y_min2 = ",y_min2)
    #print("y_max2 = ",y_max2)

#Отображение исходных данных и прямой МНК:
show_graf(x_train, y_train, k, b)
#Отображение гистограммы известного пассажиропотока:
show_hist(y_train)

#Отображение гистограммы известных расстояний точек до прямой:
#show_hist(h)

#Прогноз бутстрап методом, где h - исходные расстояния, number - количество
генерируемых выборок
# для нахождения среднего значения выборки, n_samples - размер выборки:
bootstrap_forecast(h, number, n_samples)

y_pred = []
#Прогноз будущих цен для определенного месяца(вместо 2020 - прогнозируемый
год, на выходе интервал возможных значений прогноза
for year in x_test:
    forecast(year, h_min, h_max, y_pred)

print(f'Прогнозируемый пассажиропоток за годы от {x_test[0]} до
{x_test[len(x_test)-1]} равен', y_pred, sep='\n')
print(f"Реальный пассажиропоток за годы {x_test[0]} до {x_test[len(x_test)-
1]} равен ", y_test)

# Сюжетные выходы
#Красным - прогнозируемый результат методом �'утстреп
plt.plot(x_test, y_pred, color='red', linewidth=3)
#Синим - настоящие результаты в тестовой выборке
plt.plot(x_test, y_test, color='blue', linewidth=3)
#�-еленым - тренировочная выборка
plt.plot(x_train, y_train, color='green', linewidth=3)
plt.show()
```

https://github.com/romanponomarew/Machine-learning/blob/master/Butstrep(ML)2.py

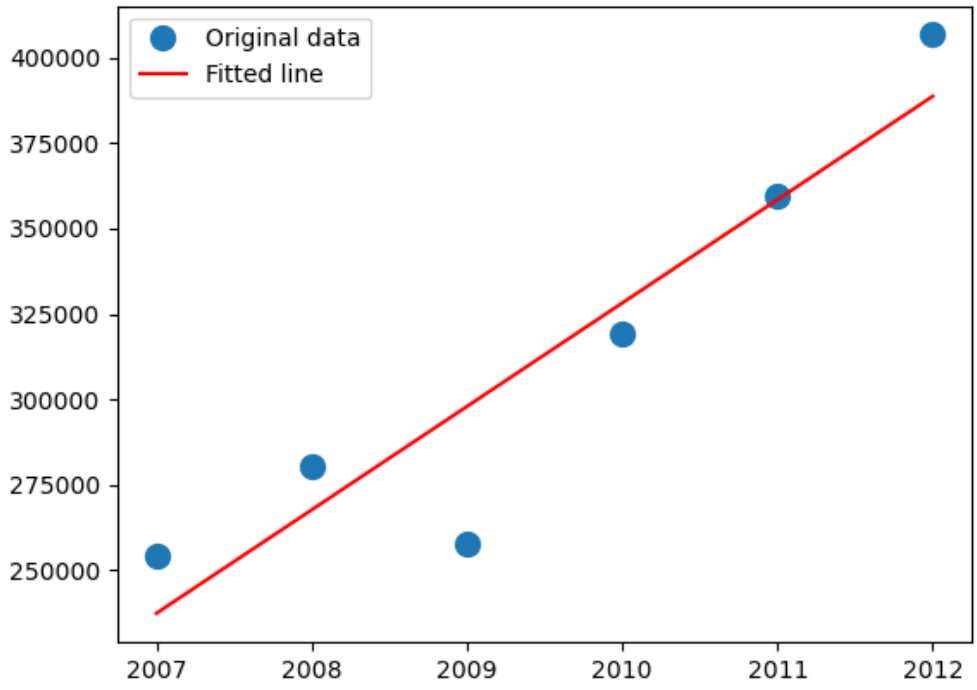

Рисунок 5 – МНК

Выявление линейной зависимости данных обучающей выборки с помощью метода МНК отображено на рисунке 5.

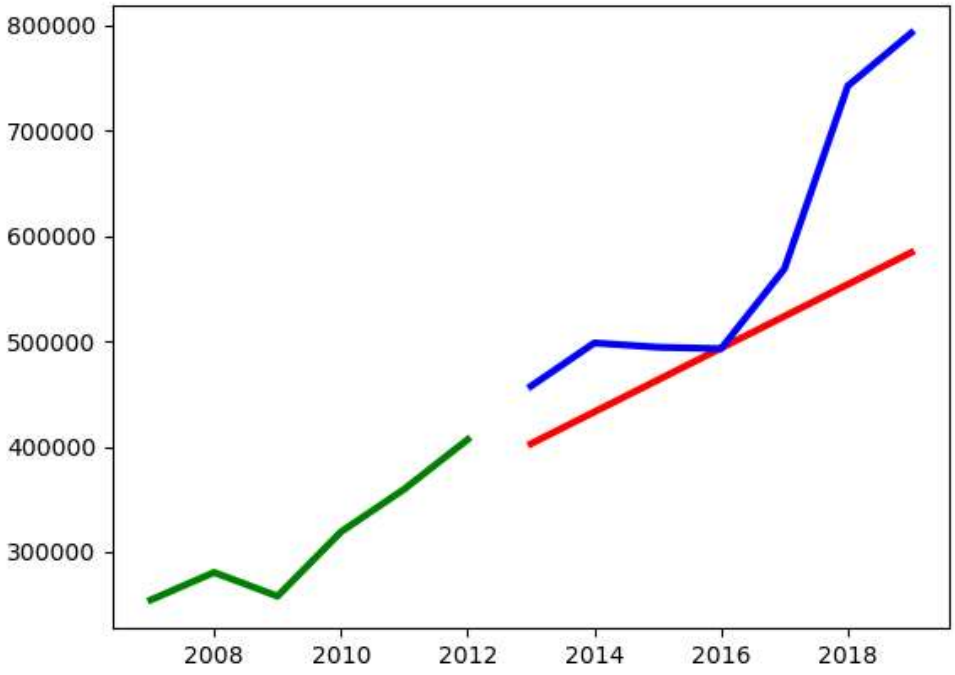

Рисунок 6 - Бутстреп метод

Анализ результатов, полученных с помощью метода бутстреп, отображенных на рисунке 6(зеленый цвет – тренировочная выборка, синий

цвет – тестовая выборка, красный цвет – спрогнозированная линейная зависимость) позволяет сделать вывод о том, что прогноз реализуемый в данном методе также опирается на выявление линейной зависимости. Именно этот факт не позволяет учитывать сильную изменчивость данных, таких как выбросы и потому прогноз говорит лишь о примерной тенденции изменения данных – возрастание либо убывание.

Вывод

Бутстреп метод, также как и метод простой линейной регрессии не следует применять для прогнозирования точных значений, ввиду сильных отличий полученных данных от реальных. Существенным недостатком методов является то, что они в своих прогнозах опираются на линейную зависимость изменения данных, что редко встречается на практике.

При этом, если все же сравнивать точность прогнозов двух методов, можно прийти к выводу, что бутстреп метод дает менее ошибочный результат.

Таким образом, данная работа позволяет сделать вывод, что рассмотренные методы подходят для прогнозирования общей тенденции изменения данных ввиду простоты алгоритмов. При этом, алгоритмы не следует применять для точного прогноза данных по причине большой погрешности получаемых данных.

Список литературы

1. https://habr.com/ru/company/ods/blog/324402/
2. https://neurohive.io/ru/osnovy-data-science/ansamblevye-metody-begging-busting-i-steking/
3. https://habr.com/ru/company/ods/blog/322076/
4. https://proglib.io/p/ml-regression/