Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

УДК 004.622

Голубко
Дмитрий Владимирович

Система менеджмента недвижимости и анализа цен на рынке

ДИССЕРТАЦИЯ
на соискание академической степени
магистра наук

по специальности 1-40 80 05 — Программная инженерия

Научный руководитель Смолякова О. Г.
к.т.н., доцент

Минск 2021

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ

МНК – Метод наименьших квадратов

ИНС – Искусственная нейронная сеть

MSE – Mean squared error, среднеквадратическая ошибка

MLP – Multilayer perceptron – Многослойный перцептрон

RBF network – Radial basis function network – Сеть радиально-базисных функций

SSR – Sum of squared residuals – Сумма квадратов остатков

SST – Sum of squares total – Общая сумма квадратов

GRNN – General regression neural network – Генерализованная регрессионная нейронная сеть

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Цель и задачи исследования

Цель диссертационной работы – провести анализ применяемых моделей и алгоритмов, которые используются при прогнозировании цен на рынке недвижимости, проверить адекватность применения методов эконометрического анализа для оценки объектов недвижимости и построение на их основе модели стоимости, применить существующие алгоритмы на подготовленной выборке данных. На основе проведенного анализа выполнить сравнение алгоритмов, определить наиболее эффективный из них. Результаты исследования могут быть полезны для прогнозирования ценообразования на рынке недвижимости, а также при оценке стоимости объектов недвижимости.

Для достижения поставленной цели необходимо решить следующие задачи:

а) Провести анализ применяемых моделей и алгоритмов, которые используются при прогнозировании цен на рынке недвижимости.

б) Реализовать предложенные модели и провести экспериментальные исследования на основе накопленных данных.

в) На основе сравнительного анализа выбрать более эффективный и достоверный.

Объектом исследования выступает процесс формирования цен на недвижимость.

Предметом исследования являются модели и алгоритмы, которые используются для анализа процесса формирования цен.

Основной *гипотезой*, положенной в основу диссертационной работы, является сравнение результатов работы выбранных алгоритмов и выявление более эффективного способа оценки стоимости недвижимости.

Структура и объем диссертации

Диссертация состоит из введения, общей характеристики работы, двух глав, заключения, списка использованных источников, списка публикаций автора и приложений. В первой главе представлен анализ существующих применяемых моделей и алгоритмов. Вторая глава посвящена проведению эксперименального исследования, реализацией выбранных алгоритмов анализа, проведению их сравнительного анализа.

ВВЕДЕНИЕ

Переход к рыночным отношениям в экономике и научно-технический прогресс чрезвычайно ускорили темпы внедрения во все сферы социально-экономической жизни общества последних научных разработок в области информационных технологий.

Рынок недвижимости представляет собой механизм, обслуживающий и регулирующий отношения по купле, продаже и аренде недвижимости на основе спроса и предложения. В мировой практике можно выделить следующие типы рынков недвижимости:

– рынок жилой недвижимости;

– рынок коммерческой недвижимости, приносящей доход ее владельцу (офисные, торговые, производственные, складские помещения);

– рынок земельных участков.

Рынок недвижимости делится на первичный и вторичный. Объектом сделок на первичном рынке является новая недвижимость, т.е. только что построенные дома, квартиры, офисные и другие помещения. Их могут продавать застройщики, инвесторы, финансировавшие строительство. На вторичном рынке предоставлено жилье и помещения, которыми уже пользовались по основному назначению. Первичный рынок отражает объемы созданной жилой недвижимости, а объем вторичного рынка определяется другими факторами:

– изменением благосостояния населения;

– доходностью различных инвестиционных объектов;

– мобильностью трудовых ресурсов;

– событиями человеческой жизни (свадьба, развод, рождение ребенка в семье, смена места жительства и др.).

Объекты недвижимости занимают значительную часть ресурсов экономики любой страны. Оценка стоимости – длительный и сложный процесс установления денежного эквивалента стоимости объекта недвижимости. Она требует высокой квалификации оценщика, владеющего методами и инструментарием оценочной деятельности, знающего состояние рынка недвижимости и особенно нужного сегмента, детального значения правовых особенностей сделок с недвижимостью и др [1]. Практика показывает, что для оценки стоимости объекта недвижимости, специалисту требуется значительное время.

С развитием теоретических подходов для создания адекватных моделей поведения рынка недвижимости в западных странах и США одновременно происходило активное внедрение новых интеллектуальных компьютерных

технологий в практику принятия финансовых и инвестиционных решений. Вначале в виде экспертных систем и баз знаний, а затем с конца 80-х - нейросетевых технологий, которые являются адекватным аппаратом для решения задач прогнозирования.

Начало исследования методов обработки информации, называемых сегодня нейросетевыми, было положено несколько десятилетий назад. С течением времени интерес к нейросетевым технологиям то ослабевал, то вновь возрождался. Такое непостоянство напрямую связано с практическими результатами проводимых исследований.

Цена на жилье - вопрос крайне сложный, изучение основных факторы влияния и выяснение правил изменения имеют важное теоретическое и практическое значение для способствования устойчивому и здоровому развитию рынка жилой недвижимости. Автоматизация позволит ускорить процесс принятия решения, учесть большее количество факторов и снизить уровень субъективности.

1 АНАЛИЗ СУЩЕСТВУЮЩИХ АЛГОРИТМОВ

1.1 Регресионный анализ

В статистическом моделировании регрессионный анализ – это набор статистических процедур для изучения зависимостей между случайными переменными. Он включает в себя множество методов моделирования и анализа взаимосвязей между зависимой переменной и одной или несколькими независимыми переменными, называемых также предикторами или регрессорами.

Регрессионный анализ помогает понять, как значение зависимой переменной изменяется при изменении одной из независимых переменных, в то время как другие независимые переменные остаются фиксированными.

Чаще всего в регрессионном анализе оценивает условное математическое ожидание зависимой переменной с учетом значений, принимаемых независимыми переменными. Во всех случаях оцениваться функция математического ожидания зависимой переменной от независимых переменных, называемая функцией регрессии.

Регрессионный анализ широко используется для численного предсказания, классификации и прогнозирования, где его применение существенно перекрывается с областью машинного обучения.

В настоящее время разработано много методов регрессионного анализа. Наиболее популярными из них являются простая и множественная линейная регрессия, среднеквадратическая и логистическая регрессия. Эти модели, являются параметрическими в том смысле, что функция регрессии определяется конечным числом неизвестных параметров, которые оцениваются на основе данных.

В математической статистике линейная регрессия представляет собой метод аппроксимации зависимостей между входными и выходными переменными на основе линейной модели. Является частью более широкой статистической методики, называемой регрессионным анализом.

В регрессионном анализе входные (независимые) переменные называются также предикторными переменными или регрессорами, а зависимые переменные — критериальными.

Если рассматривается зависимость между одной входной и одной выходной переменными, то имеет место простая линейная регрессия. Для этого определяется уравнение регрессии и строится соответствующая прямая, известная как линия регрессии(рис. 1.1).

$$y = ax + b \tag{1.1}$$

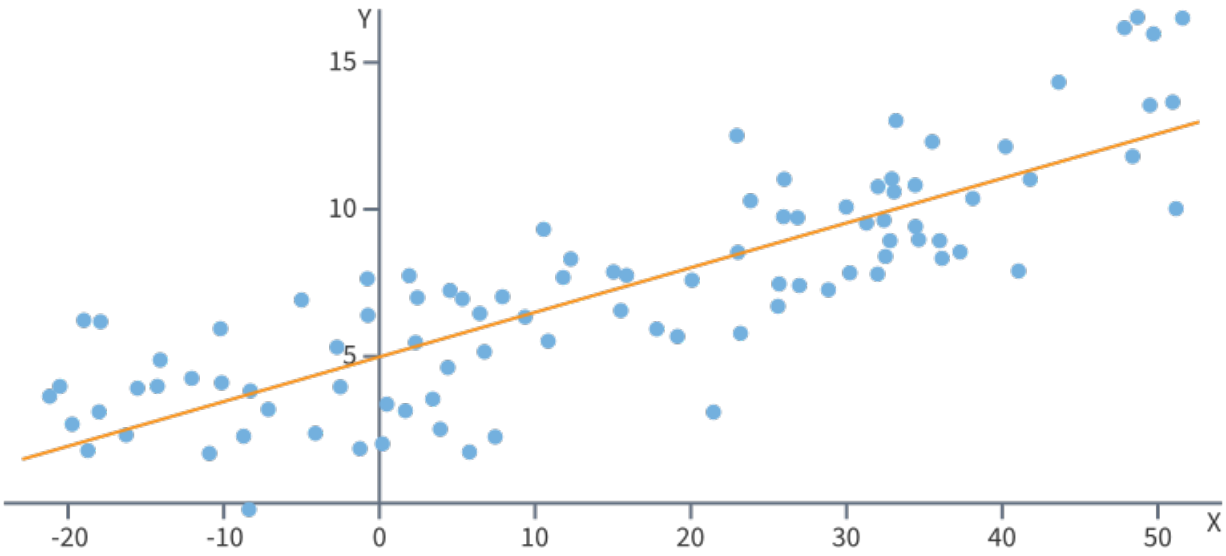

Рисунок 1.1 – Линия регрессии

Коэффициенты *a* и *b*, называемые также параметрами модели, определяются таким образом, чтобы сумма квадратов отклонений точек, соответствующих реальным наблюдениям данных, от линии регрессии была бы минимальной. Коэффициенты обычно оцениваются методом наименьших квадратов.

Метод наименьших квадратов (МНК) — математический подход для оценки параметров моделей (например, регрессионной) на основании экспериментальных данных, содержащих случайные ошибки.

Если данные известны с некоторой погрешностью, то вместо неизвестного точного значения параметра модели используется приближенное. Поэтому параметры модели должны быть рассчитаны так, чтобы минимизировать разницу между экспериментальными данными и теоретическими (вычисленными при помощи предложенной модели).

Мерой рассогласования между фактическими значениями и значениями, оцененными моделью в методе наименьших квадратов, служит сумма квадратов разностей между ними, т.е.:

$$\sum_{i=1}^{N}(\mathrm{y}' - \mathrm{y})^2 \qquad (1.2)$$

где y' — оценка, полученная с помощью модели;

y — фактическое наблюдаемое значение.

Очевидно, что лучшей будет та модель, которая минимизирует данную сумму.

Важнейшим применением МНК в анализе данных является линейная регрессия, где параметры регрессионной модели вычисляются таким образом, чтобы сумма квадратов расстояний от линии регрессии до фактических значений данных была минимальной [2].

Если ищется зависимость между несколькими входными и одной выходной переменными, то имеет место множественная линейная регрессия. Соответствующее уравнение имеет вид:

$$\mathrm{Y} = \mathrm{b}_0 + \mathrm{b}_1\mathrm{x}_1 + \mathrm{b}_2\mathrm{x}_2 + \cdots + \mathrm{b}_n\mathrm{x}_n \tag{1.3}$$

где n — число входных переменных.

Очевидно, что в данном случае модель будет описываться не прямой, а гиперплоскостью. Коэффициенты уравнения множественной линейной регрессии подбираются так, чтобы минимизировать сумму квадратов отклонения реальных точек данных от этой гиперплоскости.

Преимущество множественной линейной регрессии по сравнению с простой заключается в том, что использование в модели нескольких входных переменных позволяет увеличить долю объяснённой дисперсии выходной переменной, и таким образом улучшить соответствие модели данным. Т.е. при добавлении в модель каждой новой переменной коэффициент детерминации растёт.

Коэффициент детерминации – Статистический показатель, отражающий объясняющую способность регрессии *a:X -> Y* и равный отношению суммы квадратов регрессии SSR к общей вариации SST:

$$\mathrm{r}^2 = \frac{SSR}{SST} = \frac{\sum_i(\mathrm{a}(\mathrm{x}_i) - \bar{y})^2}{\sum_i(\mathrm{y}_i - \bar{y})^2} \tag{1.4}$$

где $\mathrm{x}_l = (\mathrm{x}_i, \mathrm{y}_i)_{i=1}^{l}$ — набор данных из *l* наблюдений.
x_i — вектор признаков *i*-го наблюдения,
$\mathrm{y}_i \in \mathrm{Y}$ — y_i принадлежит *Y*;

Данный показатель является статистической мерой согласия, с помощью которой можно определить, насколько уравнение регрессии соответствует реальным данным.

Коэффициент детерминации изменяется в диапазоне от 0 до 1. Если он равен 0, это означает, что связь между переменными регрессионной мо-

дели отсутствует и вместо нее для оценки значения выходной переменной можно использовать простое среднее ее наблюдаемых значений. Напротив, если коэффициент детерминации равен 1, это соответствует идеальной модели, когда все точки наблюдений лежат точно на линии регрессии, т.е. сумма квадратов их отклонений равна 0.

На практике, если коэффициент детерминации близок к 1, это указывает на то, что модель работает очень хорошо (имеет высокую значимость), а если к 0, то это означает низкую значимость модели, когда входная переменная плохо «объясняет» поведение выходной, т.е. линейная зависимость между ними отсутствует. Очевидно, что такая модель будет иметь низкую эффективность.

В некоторых случаях коэффициент детерминации может принимать небольшие отрицательные значения, если модель получилась «бесполезной» и ее предсказания хуже, чем оценки на основе среднего значения [3].

Линейная регрессия была первым видом регрессионного анализа, который был тщательно изучен и начал широко использоваться в практических приложениях. Это связано с тем, что в линейных моделях оценивание параметров проще, а также с тем, что статистические свойства полученных оценок легче определить.

Линейная регрессия имеет много практических применений. Большинство приложений попадают в одну из двух широких категорий:

– Если целью является прогнозирование, линейную регрессию можно использовать для подгонки модели к наблюдаемому набору данных.

– Если цель заключается в том, чтобы объяснить изменчивость выходной переменной, можно применить линейный регрессионный анализ для количественной оценки силы взаимосвязи между выходной и входными переменными.

Среднеквадратическая регрессия – разновидность регрессии, где при определении параметров модели используется обобщение метода наименьших квадратов (МНК) – метод наименьших средних квадратов.

Иными словами, в процессе подгонки модели к данным минимизируется не сумма квадратов остатков регрессии, а их средний квадрат:

$$E[(Y - F(X))^2] \tag{1.5}$$

где E — операция усреднения,
Y — зависимая переменная,
X — вектор независимых переменных.

Это становится возможным благодаря тому, что МНК допускает широкое обобщение, когда вместо минимизации суммы квадратов остатков можно минимизировать их некоторую положительно определённую квадратичную форму.

Смысл данного подхода заключается в том, что к результатам классического МНК (т.е. квадратам остатков) применяется дополнительное линейное преобразование - усреднение. Как известно, одним из предположений регрессии является предположение о нормальности остатков, которое в практических случаях не соблюдается. Усреднение позволяет снизить степень влияния отклонения остатков от нормального распределения на качество построенной модели.

Метод наименьших средних квадратов был сформулирован Бернардом Уидроу и Тедом Хоффом в 1960 году и применён при обучении нейронных сетей с помощью алгоритма обратного распространения ошибки.

В математическом и статистическом моделировании зависимой (выходной) называется переменная модели, которая зависит от входных переменных и случайных факторов, воздействующих на моделируемый процесс или объект.

Выходная переменная представляет результаты работы модели. Она изменяется (варьирует) под воздействием изменения входной переменной и случайных факторов. Изучение изменчивости выходной переменной при изменении входной и является целью моделирования.

В статистическом моделировании и машинном обучении независимой (входной) переменной называют величину, от которой зависит изменение выходной переменной, при этом целью построения модели является аппроксимация этой зависимости.

Аппроксимация – математический метод, в основе которого лежит замена одних математических объектов другими, близкими к исходным в том или ином смысле, но более простыми. [4]

Аппроксимация позволяет исследовать числовые характеристики и качественные свойства объекта, сводя задачу к изучению более простых или более удобных объектов (например, тех, параметры которых легко вычисляются или известны заранее).

Например, в линейной регрессии некоторая неизвестная функция, описывающая реальные наблюдения, аппроксимируется уравнением прямой, а если наблюдаемые данные носят нелинейный характер, то полиномами и т.д.

моделью, которая использует логистическую функцию для моделирования зависимости выходной переменной от набора входных в случае, когда первая является бинарной.

Это разновидность множественной регрессии, общее назначение которой состоит в анализе связи между несколькими независимыми переменными (называемыми также регрессорами или предикторами) и зависимой переменной. Регрессия в общем виде применяется, когда входные и выходная переменные непрерывные. А логистическая регрессия лучшим образом подходит, когда выходная переменная принимает только два значения.

Важность логистический регрессии обусловлена тем, что многие задачи анализа данных могут быть решены с помощью бинарной классификации или сведены к ней.

Например, с помощью логистической регрессии можно оценивать вероятность наступления (или не наступления) некоторого события: пациент болен (здоров), заемщик вернул кредит (допустил просрочку) и т.д. Благодаря этому логистическую регрессию можно рассматривать как мощный инструмент поддержки принятия решений.

Как известно, все регрессионные модели могут быть записаны в виде формулы:

$$y = F(x_1, x_2, \cdots, x_n) \tag{1.6}$$

Например, если рассматривается исход по займу, задается переменная y со значениями 1 и 0, где 1 означает, что соответствующий заемщик расплатился по кредиту, а 0 — что имел место дефолт.

Однако здесь возникает проблема: множественная регрессия не «знает», что переменная отклика бинарная по своей природе. Это неизбежно приведет к модели с предсказываемыми значениями большими 1 и меньшими 0. Но такие значения вообще не допустимы для первоначальной задачи. Таким образом, множественная регрессия просто игнорирует ограничения на диапазон значений для y.

Для решения проблемы задача регрессии может быть сформулирована иначе: вместо предсказания бинарной переменной мы предсказываем непрерывную переменную со значениями на отрезке [0,1] при любых значениях независимых переменных. Это достигается применением следующего регрессионного уравнения (логит-преобразование):

$$p = \frac{1}{1 + e^{-y}} \tag{1.7}$$

где p — вероятность того, что произойдет интересующее событие,
e — основание натуральных логарифмов 2,71…,
y — стандартное уравнение регрессии.

Зависимость, связывающая вероятность события и величину y, показана на следующем графике:

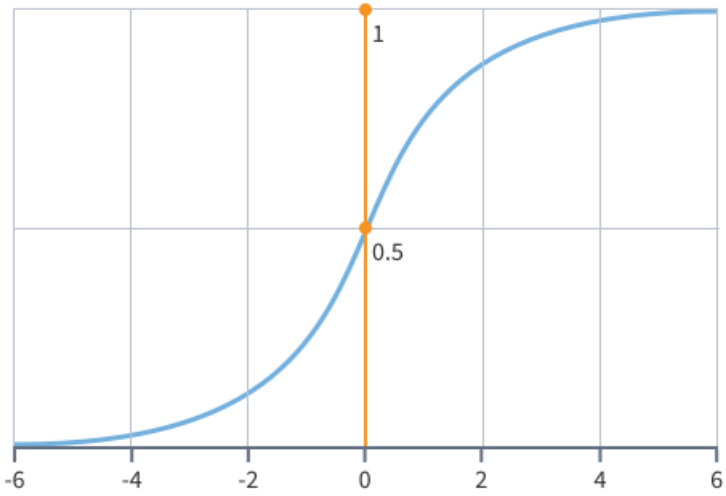

Рисунок 1.2 – Зависимость, связывающая вероятность события и величину y

Преобразование вида:

$$P' = \log_e(\frac{P}{1-P}) \tag{1.8}$$

называют логистическим, или логит-преобразованием.

Существует несколько способов нахождения коэффициентов логистической регрессии. На практике часто используют метод максимального правдоподобия. Он применяется в статистике для получения оценок параметров генеральной совокупности по выборочным данным.

Логистическая регрессия является традиционным статистическим инструментом для расчета коэффициентов (баллов) скоринговой карты на основе накопленной кредитной истории.

Генеральная совокупность — это совокупность всех объектов или наблюдений, относительно которых исследователь намерен делать выводы при решении конкретной задачи. В ее состав включаются все объекты, которые подлежат изучению.

Объем генеральной совокупности может быть очень велик, и на практике рассмотреть все ее элементы не представляется возможным. Поэтому

обычно из генеральной совокупности извлекаются выборки, на основе анализа которых аналитик пытается сделать вывод о свойствах всей совокупности, скрытых в ней закономерностях, действующих правилах и т.д. При этом выборки должны быть репрезентативными.

Регрессионный анализ является одним из наиболее распространенных методов обработки результатов экспериментов при изучении зависимостей в естественных науках, экономике, технике и др. областях.

В аналитических технологиях Data Mining элементы регрессионного анализа широко используются для решения задач прогнозирования, оценивания, классификации, выявления зависимостей между признаками.

Основы регрессионного анализа были заложены А. Лежандром и Карлом Гауссом в их работах по методу наименьших квадратов в начале 19 в. [5]

Исходя из проведенного анализа способов применения различных методов регрессии можно сделать вывод, что наиболее подходящим способом для анализа процесса формирования цен на недвижимость является множественный линейный регресионный анализ.

Данное утверждение подтверждается анализом статьи [6].

В целях оценки недвижимости может применяться либо многофакторный, либо однофакторный регрессионный анализ. В первом случае строится множественная регрессионная модель, описывающая зависимость стоимости оцениваемого объекта от нескольких независимых определяющих факторов, значения которых определяются из анализа рыночных данных. Этими факторами могут быть как физические характеристики объекта (площадь, качество отделки и т.п.), так и характеристики его местоположения (удаленность от транспортных магистралей, экологическая обстановка и т.п.).

При однофакторном регрессионном анализе рассматривается зависимость переменной – стоимости единицы сравнения – от одной независимой (контролируемой) переменной. Значения остальных независимых переменных считаются фиксированными. В качестве независимой переменной X обычно используется показатель «общая площадь», за зависимую переменную Y принимается показатель «стоимость 1м кв. общей площади».

В случае, когда рассматривается зависимость между одной зависимой переменной Y и несколькими независимыми переменными X1, X2, ... ,Xn говорят о множественной линейной регрессии.

1.2 Нейросетевой анализ

Нейронная сеть (также искусственная нейронная сеть, ИНС) — математическая модель, а также её программное или аппаратное воплощение,

построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. Это понятие возникло при изучении процессов, протекающих в мозге, и при попытке смоделировать эти процессы.

Нейронные сети используются для решения сложных задач, которые требуют аналитических вычислений подобных тем, что делает человеческий мозг. Самыми распространенными применениями нейронных сетей является:

– Классификация — распределение данных по параметрам. Например, на вход дается набор людей и нужно решить, кому из них давать кредит, а кому нет. Эту работу может сделать нейронная сеть, анализируя такую информацию как: возраст, платежеспособность, кредитная история и тд.

– Предсказание — возможность предсказывать следующий шаг. Например, рост или падение акций, основываясь на ситуации на фондовом рынке.

– Распознавание — в настоящее время, самое широкое применение нейронных сетей. Используется в Google, когда вы ищете фото или в камерах телефонов, когда оно определяет положение вашего лица и выделяет его и многое другое.

Классификация искусственных нейронных сетей представлена на рисунке 1.3

Задачи прогнозирования (предсказания) можно разбить на два основных класса: классификация и регрессия.

В задачах классификации нужно бывает определить, к какому из нескольких заданных классов принадлежит данный входной набор. Примерами могут служить предоставление кредита (относится ли данное лицо к группе высокого или низкого кредитного риска), диагностика раковых заболеваний (опухоль, чисто), распознавание подписи (поддельная, подлинная). Во всех этих случаях, очевидно, на выходе требуется всего одна номинальная переменная. Чаще всего (как в этих примерах) задачи классификации бывают двузначными, хотя встречаются и задачи с несколькими возможными состояниями.

В задачах регрессии требуется предсказать значение переменной, принимающей (как правило) непрерывные числовые значения: завтрашнюю цену акций, расход топлива в автомобиле, прибыли в следующем году и т.п.. В таких случаях в качестве выходной требуется одна числовая переменная.

Исскусственные нейронные сети обычно состоят из нейронов и связывающих их синапсов.

Нейрон — это вычислительная единица, которая получает информацию, производит над ней простые вычисления и передает ее дальше. Они де-

Рисунок 1.3 – Классификация нейронных сетей

лятся на три основных типа: входной (слева), скрытый (посередине) и выходной (справа). Также есть нейрон смещения и контекстный нейрон. В том случае, когда нейросеть состоит из большого количества нейронов, вводят термин слоя. Соответственно, есть входной слой, который получает информацию, *n* скрытых слоев (обычно их не больше 3), которые ее обрабатывают и выходной слой, который выводит результат. У каждого из нейронов есть 2 основных параметра: входные данные (input data) и выходные данные (output

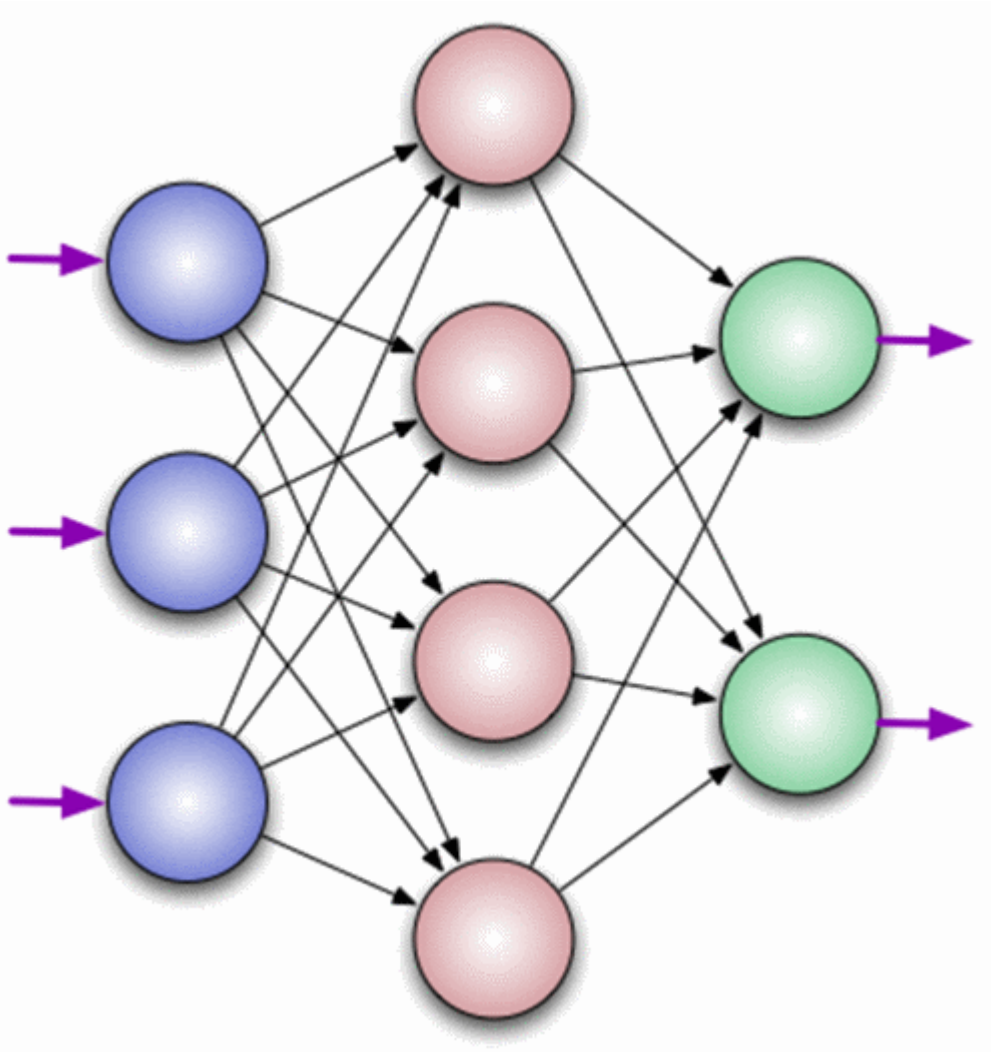

Рисунок 1.4 – Виды нейронов

data). В случае входного нейрона: input=output. В остальных, в поле input попадает суммарная информация всех нейронов с предыдущего слоя, после чего, она нормализуется, с помощью функции активации и попадает в поле output [7].

Нейрон смещения или bias нейрон — это третий вид нейронов, используемый в большинстве нейросетей. Особенность этого типа нейронов заключается в том, что его вход и выход всегда равняются 1 и они никогда не имеют входных синапсов. Нейроны смещения могут, либо присутствовать в нейронной сети по одному на слое, либо полностью отсутствовать, 50/50 быть не может (красным на схеме обозначены веса и нейроны которые размещать нельзя). Соединения у нейронов смещения такие же, как у обычных нейронов — со всеми нейронами следующего уровня, за исключением того, что синапсов между двумя bias нейронами быть не может. Следовательно, их можно размещать на входном слое и всех скрытых слоях, но никак не на выходном слое, так как им попросту не с чем будет формировать связь.

Нейрон смещения нужен для того, чтобы иметь возможность получать выходной результат, путем сдвига графика функции активации вправо или влево.

Также нейроны смещения помогают в том случае, когда все входные нейроны получают на вход 0 и независимо от того какие у них веса, они все передадут на следующий слой 0, но не в случае присутствия нейрона смещения.

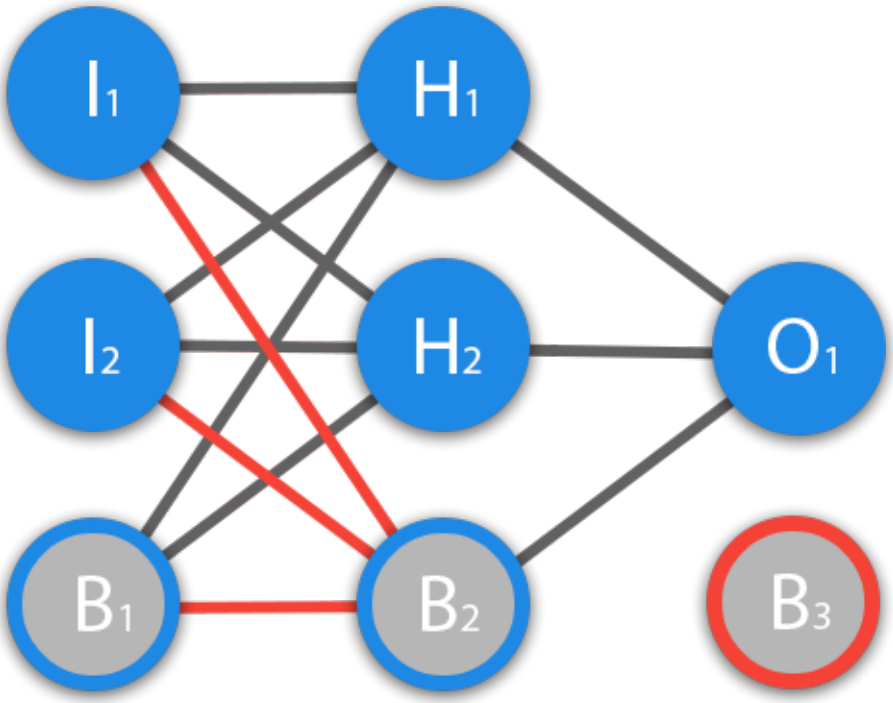

Рисунок 1.5 – Нейрон смещения

Синапс – это связь между двумя нейронами. У синапсов есть 1 параметр — вес. Благодаря ему, входная информация изменяется, когда передается от одного нейрона к другому. Допустим, есть 3 нейрона, которые передают информацию следующему. Тогда у нас есть 3 веса, соответствующие каждому из этих нейронов. У того нейрона, у которого вес будет больше, та информация и будет доминирующей в следующем нейроне (пример — смешение цветов). На самом деле, совокупность весов нейронной сети или матрица весов — это своеобразный мозг всей системы. Именно благодаря этим весам, входная информация обрабатывается и превращается в результат.

Функция активации – это способ нормализации входных данных. То есть, если на входе у вас будет большое число, пропустив его через функцию активации, вы получите выход в нужном вам диапазоне. Самые основные функции активации: Линейная, Сигмоид (Логистическая) и Гиперболический тангенс. Главные их отличия — это диапазон значений.

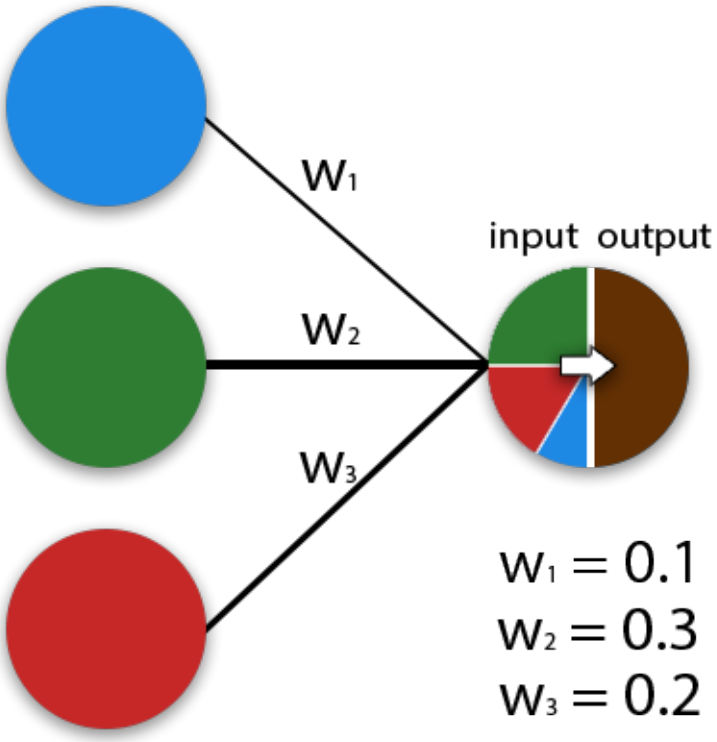

Рисунок 1.6 – Синапсы для связи нейронов

Линейная функция почти никогда не используется, за исключением случаев, когда нужно протестировать нейронную сеть или передать значение без преобразований.

$$f(x) = x$$

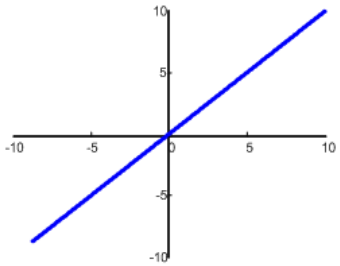

Рисунок 1.7 – Линейная функция активации

Сигмоид – это самая распространенная функция активации, ее диапазон значений [0,1]. Именно на ней показано большинство примеров в сети, также ее иногда называют логистической функцией. Соответственно, если в вашем случае присутствуют отрицательные значения (например, акции мо-

гут идти не только вверх, но и вниз), то вам понадобиться функция которая захватывает и отрицательные значения.

$$f(x) = \frac{1}{1+e^{-x}}$$

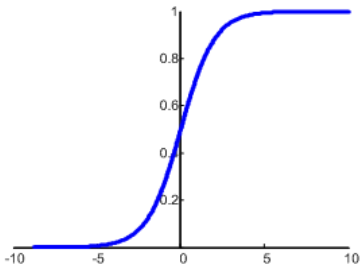

Рисунок 1.8 – Сигмоид

Имеет смысл использовать гиперболический тангенс, только тогда, когда ваши значения могут быть и отрицательными, и положительными, так как диапазон функции [-1,1]. Использовать эту функцию только с положительными значениями нецелесообразно так как это значительно ухудшит результаты вашей нейросети.

$$f(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

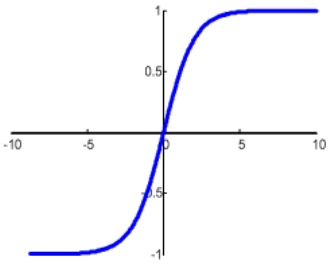

Рисунок 1.9 – Гиперболический тангенс

Ошибка — это процентная величина, отражающая расхождение между ожидаемым и полученным ответами. Ошибка формируется каждую эпоху и должна идти на спад. Ошибку можно вычислить разными путями, например: Mean Squared Error (далее MSE), Root MSE и Arctan. Здесь нет какого-либо

ограничения на использование, как в функции активации, и можно использовать любой метод, который будет приносить вам наилучший результат. Стоит лишь учитывать, что каждый метод считает ошибки по разному. У Arctan, ошибка, почти всегда, будет больше, так как он работает по принципу: чем больше разница, тем больше ошибка. У Root MSE будет наименьшая ошибка, поэтому, чаще всего, используют MSE, которая сохраняет баланс в вычислении ошибки.

Обучение искусственных нейронных сетей происходит с помощью заранее подготовленной выборки данных (тренировочный сет).

Общее количество тренировочных сетов, пройденных нейронной сетью, называется итерацией.

Эпохой называется количество пройденных наборов тренировочных сетов. Чем больше эпоха, тем лучше натренирована сеть и соответственно, ее результат.

Для получения корректных результатов работы искусственной нейронной сети, её необходимо обучить. Существует несколько методов обучения нейронных сетей, самые распространенные из них: метод обратного распространения, метод упругого распространения, генетический алгоритм.

Метод обратного распространения, использует алгоритм градиентного спуска.

Применение алгоритма обратного распространения ошибки — один из известных методов, используемых для глубокого обучения нейронных сетей прямого распространения (такие сети ещё называют многослойными персептронами). Этот метод относят к методу обучения с учителем, поэтому требуется задавать в обучающих примерах целевые значения.

Сегодня нейронные сети прямого распространения используются для решения множества сложных задач. Если говорить об обучении нейронных сетей методом обратного распространения, то тут пользуются двумя проходами по всем слоям нейросети: прямым и обратным. При выполнении прямого прохода осуществляется подача входного вектора на входной слой сети, после чего происходит распространение по нейронной сети от слоя к слою. В итоге должна осуществляться генерация набора выходных сигналов — именно он, по сути, является реакцией нейронной сети на этот входной образ. При прямом проходе все синаптические веса нейросети фиксированы. При обратном проходе все синаптические веса настраиваются согласно правил коррекции ошибок, когда фактический выход нейронной сети вычитается из желаемого, что приводит к формированию сигнала ошибки. Такой сигнал в дальнейшем распространяется по сети, причём направление распространения обратно направлению синаптических связей. Именно поэтому соответствую-

щий метод и называют алгоритмом с обратно распространённой ошибкой. Синаптические веса настраивают с целью наибольшего приближения выходного сигнала нейронной сети к желаемому.

Цель обучения нейросети при использовании алгоритма обратного распространения ошибки — это такая подстройка весов нейросети, которая позволит при приложении некоторого множества входов получить требуемое множество выходов нейронов (выходных нейронов). Можно назвать эти множества входов и выходов векторами. В процессе обучения предполагается, что для любого входного вектора существует целевой вектор, парный входному и задающий требуемый выход. Эту пару называют обучающей. Работая с нейросетями, мы обучаем их на многих парах.

Также можно сказать, что алгоритм использует стохастический градиентный спуск и продвигается в многомерном пространстве весов в направлении антиградиента, причём цель — это достижение минимума функции ошибки.

При практическом применении метода обучение продолжают не до максимально точной настройки нейросети на минимум функции ошибки, а пока не будет достигнуто довольно точное его приближение. С одной стороны, это даёт возможность уменьшить количество итераций обучения, с другой — избежать переобучения нейронной сети.

Для реализации метода обратного распространения ошибки необходимо выполнить следующие действия:

а) Инициализировать синаптические веса случайными маленькими значениями.

б) Выбрать из обучающего множества очередную обучающую пару; подать на вход сети входной вектор.

в) Выполнить вычисление выходных значений нейронной сети.

г) Посчитать разность между выходом нейросети и требуемым выходом (речь идёт о целевом векторе обучающей пары).

д) Скорректировать веса сети в целях минимизации ошибки.

е) Повторять для каждого вектора обучающего множества шаги б-д, пока ошибка обучения нейронной сети на всём множестве не достигнет уровня, который является приемлемым.

Сегодня существует много модификаций алгоритма обратного распространения ошибки. Возможно обучение не «по шагам» (выходная ошибка вычисляется, веса корректируются на каждом примере), а «по эпохам» в offline-режиме (изменения весовых коэффициентов происходит после подачи на вход нейросети всех примеров обучающего множества, а ошибка обучения нейронной сети усредняется по всем примерам).

Обучение «по эпохам» более устойчиво к выбросам и аномальным значениям целевой переменной благодаря усреднению ошибки по многим примерам. Зато в данном случае увеличивается вероятность «застревания» в локальных минимумах. При обучении «по шагам» такая вероятность меньше, ведь применение отдельных примеров создаёт «шум», «выталкивающий» алгоритм обратного распространения из ям градиентного рельефа.

К плюсам можно отнести простоту в реализации и устойчивость к выбросам и аномалиям в данных, и это основные преимущества. Но есть и минусы:

– неопределенно долгий процесс обучения;

– вероятность «паралича сети» (при больших значениях рабочая точка функции активации попадает в область насыщения сигмоиды, а производная величина приближается к 0, в результате чего коррекции весов почти не происходят, а процесс обучения «замирает»;

– алгоритм уязвим к попаданию в локальные минимумы функции ошибки.

Появление алгоритма стало знаковым событием и положительно отразилось на развитии нейросетей, ведь он реализует эффективный с точки зрения вычислительных процессов способ обучения многослойного персептрона. В то же самое время, было бы неправильным сказать, что алгоритм предлагает наиболее оптимальное решение всех потенциальных проблем [8].

Одним из серьезных недостатков алгоритма с обратным распространением ошибки, используемого для обучения многослойных нейронных сетей, является слишком долгий процесс обучения, что делает неприменимым использование данного алгоритма для широкого круга задач, которые требуют быстрого решения. В настоящее время известно достаточное количество алгоритмов ускоряющих процесс обучения, одним из них является метод упругого распространения ошибки, который был предложен М. Ридмиллером (M.Riedmiller) и Г. Брауном (H.Braun).

В отличие от стандартного алгоритма с обратным распространением ошибки, метод упругого распространения ошибки использует только знаки частных производных для подстройки весовых коэффициентов. Алгоритм использует так называемое «обучение по эпохам», когда коррекция весов происходит после предъявления сети всех примеров из обучающей выборки.

Для определения величины коррекции используется следующее правило

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^{+}\Delta_{ij}^{(t)}, \frac{\partial E^{(t)}}{\partial w_{ij}} \frac{\partial E^{(t-1)}}{\partial w_{ij}} > 0 \\ \eta^{-}\Delta_{ij}^{(t)}, \frac{\partial E^{(t)}}{\partial w_{ij}} \frac{\partial E^{(t-1)}}{\partial w_{ij}} < 0 \end{cases} \tag{1.9}$$

Если на текущем шаге частная производная по соответствующему весу w_{ij} поменяла свой знак, то это говорит о том, что последнее изменение было большим, и алгоритм проскочил локальный минимум (соответствующую теорему можно найти в любом учебнике по математическому анализу), и, следовательно, величину изменения необходимо уменьшить на η и вернуть предыдущее значение весового коэффициента: другими словами необходимо произвести 'откат'.

$$\Delta w_{ij}(t) = \Delta w_{ij}(t) - \Delta_{ij}^{(t-1)} \tag{1.10}$$

Если знак частной производной не изменился, то нужно увеличить величину коррекции на η^+ для достижения более быстрой сходимости. Зафиксировав множители η^- и η^+, можно отказаться от глобальных параметров настройки нейронной сети, что также можно рассматривать как преимущество рассматриваемого алгоритма перед стандартным алгоритмом.

Рекомендованные значения для η^-= 0.5, η^+= 1.2, но нет никаких ограничений на использование других значений для этих параметров.

Для того, чтобы не допустить слишком больших или малых значений весов, величину коррекции ограничивают сверху максимальным Δ_{max} и снизу минимальным Δ_{min} значениями величины коррекции, которые по умолчанию, соответственно, устанавливаются равными 50 и 1.0E-6.

Начальные значения для всех Δ_{ij} устанавливаются равными 0.1. Опять же, это следует рассматривать лишь как рекомендацию, и в практической реализации можно задать другое значение для инициализации.

Для вычисления значения коррекции весов используется следующее правило:

$$\Delta w_{ij}(t) = \begin{cases} -\Delta_{ij}^{(t)}, \frac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ +\Delta_{ij}^{(t)}, \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ 0, \frac{\partial E^{(t)}}{\partial w_{ij}} = 0 \end{cases} \tag{1.11}$$

Если производная положительна, т.е. ошибка возрастает, то весовый коэффициент уменьшается на величину коррекции, в противном случае – увеличивается.

Затем подстраиваются веса:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t) \tag{1.12}$$

Данный алгоритм сходится в 4-5 раз быстрее, чем стандартный алгоритм Backprop [9].

Нейросеть можно обучать с учителем и без.

Обучение с учителем — это тип тренировок присущий таким проблемам как регрессия и классификация. Иными словами здесь вы выступаете в роли учителя, а НС в роли ученика. Вы предоставляете входные данные и желаемый результат, то есть ученик посмотрев на входные данные поймет, что нужно стремиться к тому результату который ему предоставлен.

Обучение без учителя — этот тип обучения встречается не так часто. Здесь нет учителя, поэтому сеть не получает желаемый результат или же их количество очень мало. В основном такой вид тренировок присущ НС у которых задача состоит в группировке данных по определенным параметрам.

Существует еще такой метод, как обучение с подкреплением. Такой способ применим тогда, когда можно основываясь на результатах полученных от НС, дать ей оценку. НС поредоставляется право найти любой способ достижения цели, до тех пор пока он будет давать хороший результат. Таким способом, сеть начнет понимать чего от нее хотят добиться и пытается найти наилучший способ достижения этой цели без постоянного предоставления данных “учителем”.

Приведенный выше анализ существующих типов нейронных сетей позволяет сделать вывод, что целесообразным является использование нейронной сети для задачи прогнозирования, обучение с учителем с применением алгоритма обратного распространения ошибки.

Данное удтверждение подтверждается анализом статьи [10]. В данной статье проведено исследование и разработка методики оценки стоимости недвижимости с использованием нейросетевых технологий. Были решены задачи предварительного отбора факторов, оказывающих влияние на рыночную стоимость квартир; подготовлена обучающая выборка и определены оптимальные типы и характеристики, а также метода ее обучениия.

Обучающая выборка построена для проектирования и обучения нейронной сети «с учителем», к реализации предполагается 3 типа сетей: многослойный персептрон (MLP); сеть радиально-базисных функций (RBF); обобщенно-регрессионная нейронная сеть (GRNN).

Многослойный персептрон - это класс искусственных нейронных сетей прямого распространения, состоящих как минимум из трех слоёв: входного, скрытого и выходного. За исключением входных, все нейроны использует нелинейную функцию активации.

При обучении MLP используется обучение с учителем и алгоритм обратного распространения ошибки.

Обобщенная схема многослойного перспетрона показана на рисунке 1.10

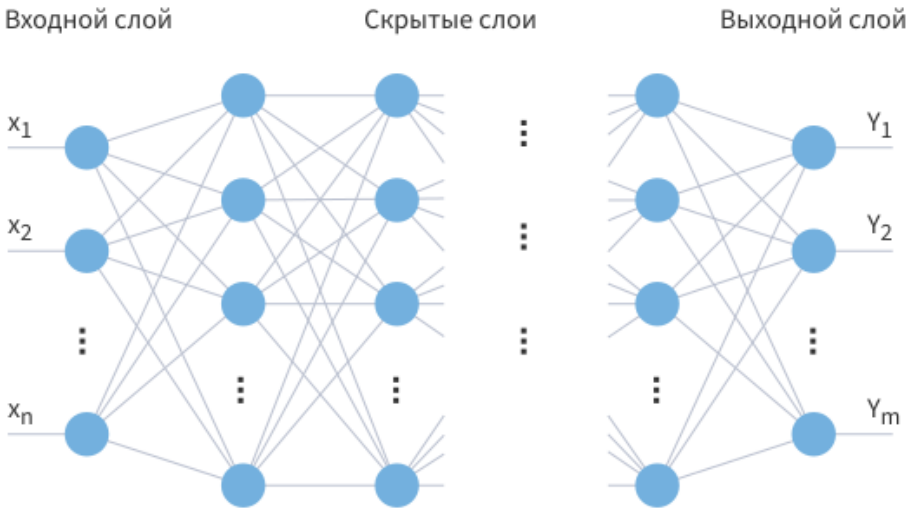

Рисунок 1.10 – Схема многослойного перспетрона

В качестве активационных функций нейронов используются сигмоидальные: логистическая или гиперболический тангенс.

MLP показали возможность находить приближённые решения для чрезвычайно сложных задач. В частности, они являются универсальным аппроксиматором функций, поэтому с успехом используются в построении регрессионных моделей. Поскольку классификацию можно рассматривать как частный случай регрессии, когда выходная переменная категориальная, на основе MLP можно строить классификаторы.

Пик популярности MLP в машинном обучении пришёлся на 1980-е годы в таких областях, как распознавание речи и изображений, системах машинного перевода. Однако позднее они столкнулись с конкуренцией с другими технологиями машинного обучения, такими, как машины опорных векторов. Интерес к многослойным персептронам вернулся благодаря успехам глубокого обучения.

Радиально-симметричные функции – простейший класс функций. В принципе, они могут быть использованы в разных моделях (линейных и нелинейных) и в разных сетях (многослойных и однослойных). Традиционно термин RBF сети ассоциируется с радиально-симметричными функциями в однослойных сетях, имеющих структуру, представленную на рисунке 1.11.

То есть, каждый из n компонентов входного вектора подается на вход m базисных функций и их выходы линейно суммируются с весами.

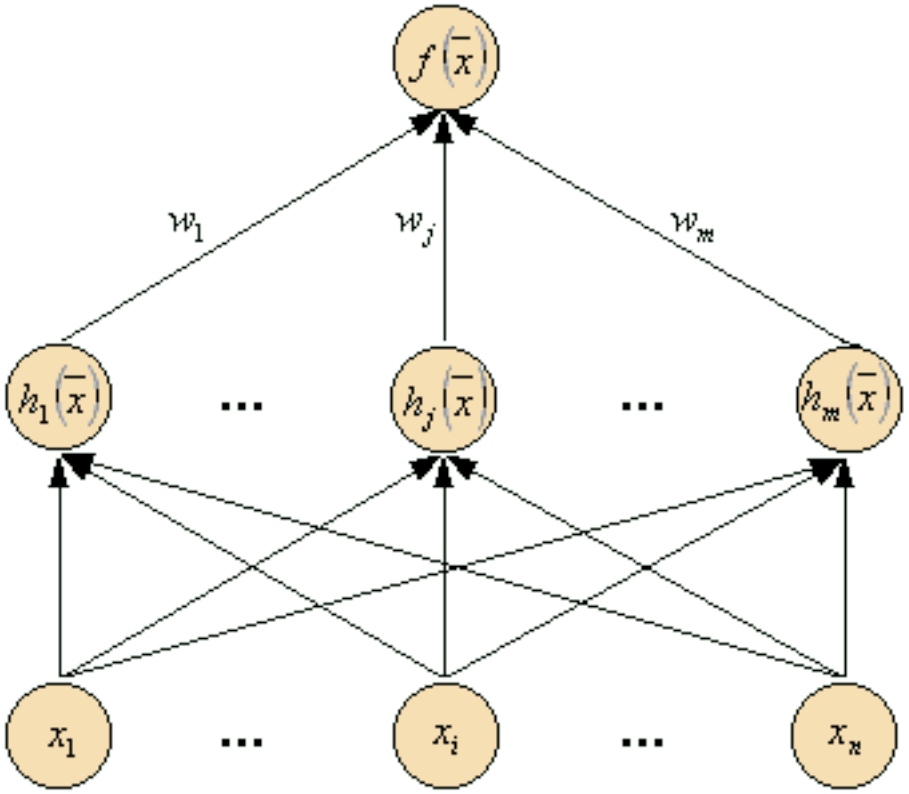

Рисунок 1.11 – Схема RBF сети

Таким образом, выход RBF сети является линейной комбинацией некоторого набора базисных функций:

$$f(\overline{x}) = \sum_{j=1}^{m} w_j h_j \tag{1.13}$$

Обобщенно-регрессионная нейронная сеть (GRNN) устроена аналогично вероятностной нейронной сети (PNN), но она предназначена для решения задач регрессии, а не классификации. Обобщенная схема GRNN сети представлена на рисунке 1.12. Как и в случае PNN-сети, в точку расположения каждого обучающего наблюдения помещается гауссова ядерная функция. Мы считаем, что каждое наблюдение свидетельствует о некоторой нашей уверенности в том, что поверхность отклика в данной точке имеет определенную высоту, и эта уверенность убывает при отходе в сторону от точки. GRNN-сеть копирует внутрь себя все обучающие наблюдения и использует их для оценки отклика в произвольной точке. Окончательная выходная оценка сети получается как взвешенное среднее выходов по всем обучающим наблюдениям, где величины весов отражают расстояние от этих наблюдений до той точки, в которой производится оценивание (и, таким образом, более близкие точки вносят больший вклад в оценку).

Первый промежуточный слой сети GRNN состоит из радиальных эле-

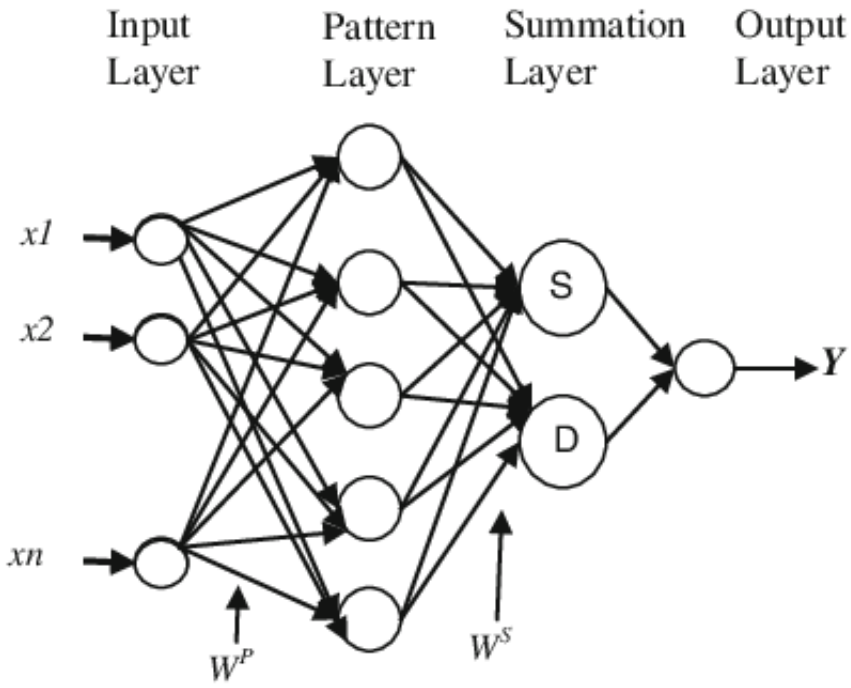

Рисунок 1.12 – Схема GRNN сети

ментов. Второй промежуточный слой содержит элементы, которые помогают оценить взвешенное среднее. Для этого используется специальная процедура. Каждый выход имеет в этом слое свой элемент, формирующий для него взвешенную сумму. Чтобы получить из взвешенной суммы взвешенное среднее, эту сумму нужно поделить на сумму весовых коэффициентов. Последнюю сумму вычисляет специальный элемент второго слоя. После этого в выходном слое производится собственно деление (с помощью специальных элементов "деления"). Таким образом, число элементов во втором промежуточном слое на единицу больше, чем в выходном слое. Как правило, в задачах регрессии требуется оценить одно выходное значение, и, соответственно, второй промежуточный слой содержит два элемента.

Можно модифицировать GRNN-сеть таким образом, чтобы радиальные элементы соответствовали не отдельным обучающим случаям, а их кластерам. Это уменьшает размеры сети и увеличивает скорость обучения. Центры для таких элементов можно выбирать с помощью любого предназначенного для этой цели алгоритма (выборки из выборки, K-средних или Кохонена).

GRNN-сеть обучается почти мгновенно, но может получиться большой и медленной (хотя здесь, в отличие от PNN, не обязательно иметь по одному радиальному элементу на каждый обучающий пример, их число все равно будет большим). Как и сеть RBF, сеть GRNN не обладает способностью экстраполировать данные.

Проведенное сравнение полученных результатов(рис. 1.13), полученных в статье [10] позволяет сделать выводы.

Тип	Обучение	Контроль	Тест
MLP			
Средняя ошибка	-7815.80	57316.31	-15922.77
Абсолютная средняя ошибка	149701.10	176512.5	203957
Коэф. регрессии	0.19	0.26	0.20
Корреляция	0.98	0.96	0.98
РБФ-сеть			
Средняя ошибка	-4.168e-09	130071	-60229.71
Абсолютная средняя ошибка	356535.5	397528.1	390934.1
Коэф. регрессии	0.5297183	0.663065	0.43
Корреляция	0.84	0.76	0.92
GRNN-сеть			
Средняя ошибка	30.27681	-22907.96	-339579
Абсолютная средняя ошибка	32448.87	282364.5	551109
Коэф. регрессии	0.11	0.57	0.81
Корреляция	0.99	0.82	0.58

Рисунок 1.13 – Сравнение полученных результатов

GRNN-сеть показала очень хорошие результаты на тестовой выборке, в то время как на тестовой выборке ее эффективность оказалась значительно ниже, чем у прочих рассмотренных сетей. Наиболее вероятным здесь событием является нерешенная проблема переобучения. То есть минимизировалась не та ошибка, которая ожидается от сети при подаче совершенно новых значений. Другими словами, у данной сети отсутствует способность обобщать результаты работы на новые наблюдения.

РБФ-сеть не продемонстрировала высоких результатов, однако несомненным ее достоинством является более высокая скорость обучения.

Многослойный персептрон является наиболее подходящим вариантом решения задачи определения стоимости жилых квартир. Полученные данные позволяют с достаточной точностью прогнозировать стоимость квартир по заданным параметрам.

Достаточно высокая точность полученных результатов является следствием тщательно сформированной обучающей выборки. Сравнение различных алгоритмов тренировки нейронной сети позволило выявить оптимальный

алгоритм для имеющегося набора данных. При этом, следует заметить, что все изучаемые алгоритмы продемонстрировали сравнительно высокую точность предсказания результата.

Рассмотрены и реализованы 3 типа сетей: многослойный персептрон (MLP); сеть радиально-базисных функций (RBF); обобщенно-регрессионная нейронная сеть (GRNN). Многослойный персептрон является наиболее подходящим вариантом решения задачи определения стоимости жилых квартир. Полученные данные позволяют с достаточной точностью прогнозировать стоимость квартир по заданным параметрам.

2 ЭКСПЕРЕМЕНТАЛЬНАЯ ЧАСТЬ

2.1 Подготовка данных

В связи с тем, что рынок вторичного жилья лучше соответствует рыночным принципам формирования цен на основе спроса и предложения, в отличие от цен, устанавливаемых компанией-застройщиком жилья на первичном рынке, определение рыночной стоимости, как наиболее вероятной цены продажи объекта недвижимости, более целесообразно провести на примере объектов недвижимости вторичного рынка жилья. При создании модели оценки жилой недвижимости в качестве входных параметров были включены факторы, представленные в таблице 2.1:

Таблица 2.1 – Характеристики недвижимости

Параметр	Описание
Building area	Общая площадь, кв.м
Carport count	Количество парковочных мест под навесом
Garage count	Количество гаражей
Open spaces count	Количество открытых парковочных мест
Property Type	Тип недвижимости(дом, таунхаус, апартаменты, итд.)
Bathrooms	Количество санузлов
Bedrooms	Количество спален
Number of sales	Количество проданных жилых объектов в данном городе за 5 лет
Average Price	Средняя цена проданных жилых объектов в данном городе за 5 лет
Median Rental Price	Медианная стоимость аренды недвижимости в данном городе за 5 лет
Change in Rental Rate	Процент изменения средней арендной платы в данном городе за 5 лет
Sold Price	Стоимость данного объекта

Как можно заметить, полученные данные являются как количественными, так и качественными. Количественные данные остаются без изменений, для качественных(тип недвижимости) были введены числовые характеристики. Исходные данные были взяты из базы проданной недвижимости в Австралии за период с 2017 по 2020 год. Всего было собрано данных о более чем 5 тысяч проданных объектов недвижимости. Данные хранятся в виде таблицы в формате, часть которой показана на рисунке 2.1

	A	B	C	D	E	F	G	H	I	J	K	L
1	building_area	carport_count	garage_count	open_spaces_count	property_type_id	bathrooms	bedrooms	number_of_sales	average_price	median_rental_price	change_in_rental_rate	sold_price
2	245.0	0	2	0	2	2	3	182	1087705	590	7	790000
3	102.0	0	2	0	1	2	3	53	1360472	615	6	751550
4	111.0	0	1	0	2	2	2	182	1087705	590	7	642000
5	330.0	0	2	1	1	3	6	38	845250	490	-6	970000
6	260.0	0	2	0	1	2	4	87	936559	490	4	1007000
7	219.0	0	2	0	1	2	4	98	741236	495	15	835000
8	295.0	0	2	0	1	3	5	98	741236	495	15	1401000
9	368.0	0	2	0	1	3	3	88	1195110	650	24	1005000
10	261.0	0	1	0	1	2	3	88	1195110	650	24	1015000
11	116.0	1	0	0	2	2	3	88	1195110	650	24	718000
12	230.0	0	2	0	1	2	3	76	645917	395	10	710000
13	318.0	0	1	0	1	2	3	197	744420	485	8	642500
14	313.0	0	2	0	1	3	4	87	936559	490	4	1512000
15	220.0	0	1	0	1	1	4	9	839000	410	15	945000
16	205.0	1	1	0	2	3	3	88	1195110	650	24	850000
17	220.0	0	2	0	1	3	3	88	1195110	650	24	1400000
18	255.0	0	2	0	1	2	3	87	936559	490	4	880000
19	161.0	0	2	0	4	2	3	5	651000	368	-3	470000
20	194.0	0	1	0	1	2	4	9	839000	410	15	778000
21	135.0	0	1	0	2	3	3	182	1087705	590	7	840000
22	286.0	0	1	0	1	2	4	88	1195110	650	24	1510000
23	266.0	0	2	0	1	3	4	182	1087705	590	7	1025000
24	277.0	0	2	0	1	1	2	88	1195110	650	24	875000
25	140.0	0	1	0	4	1	2	88	1195110	650	24	700000
26	202.0	0	1	0	1	2	4	75	813226	460	3	615000
27	302.0	0	5	0	1	3	4	53	1360472	615	6	1000000
28	280.0	0	1	2	1	3	4	98	741236	495	15	790000
29	86.0	0	2	0	4	1	2	6	605500	440	10	400500
30	143.0	0	1	0	2	2	3	30	742150	470	9	565000
31	475.0	0	3	0	1	3	5	182	1087705	590	7	1565000
32	222.0	0	2	0	1	3	5	182	1087705	590	7	1130000
33	130.0	0	1	0	4	1	2	88	1195110	650	24	690000
34	204.0	0	2	0	1	3	3	182	1087705	590	7	780000

Рисунок 2.1 – Исходные данные в csv формате

Проведен анализ полученных данных. На рисунке 2.2 приведена гистограмма площадей полученных объектов недвижимости, которая позволяет говорить что наиболее популярная недвижмиость имеет площадь от 100 до 150 квадратных метров.

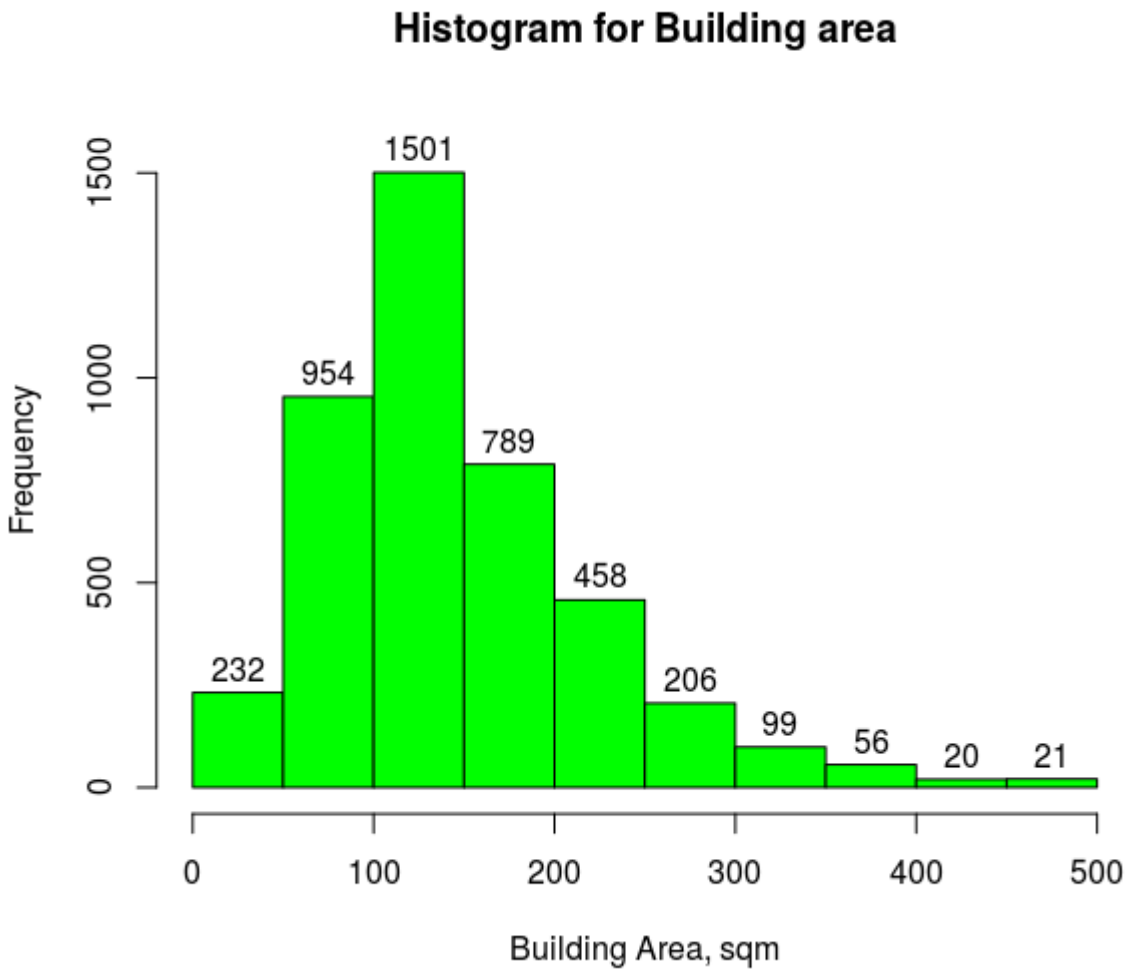

Рисунок 2.2 – Гистограмма площадей

На рисунке 2.3 приведена гистограмма душевых комнат полученных объектов недвижимости, на которой видно что подавляющее большинство домов имеют 1 или 2 душевые комнаты.

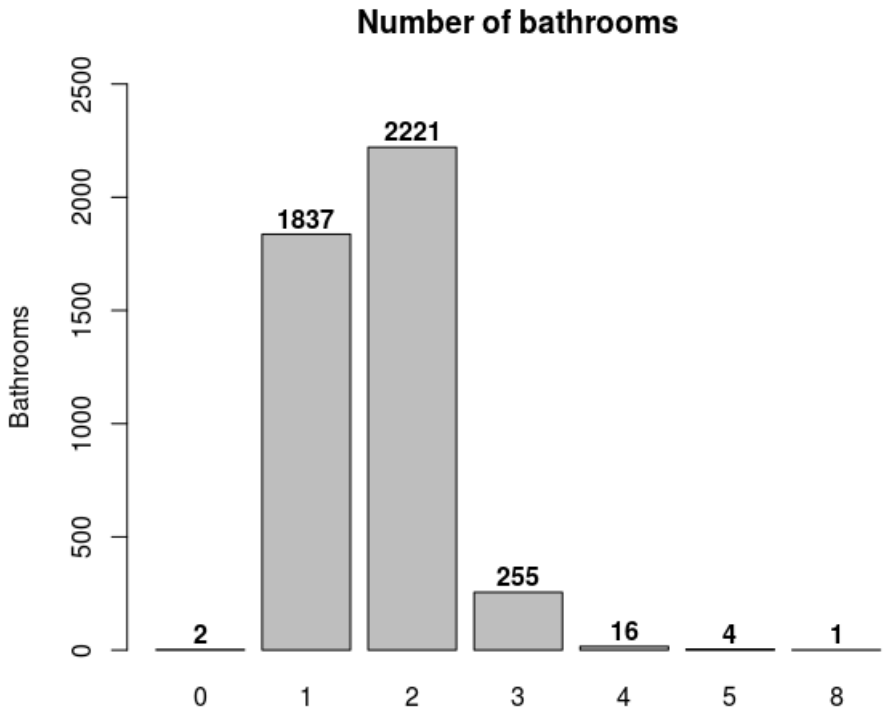

Рисунок 2.3 – Гистограмма душевых комнат

На рисунке 2.4 приведена гистограмма спален полученных объектов недвижимости, на которой видно что большинство домов от двух до четырех спален.

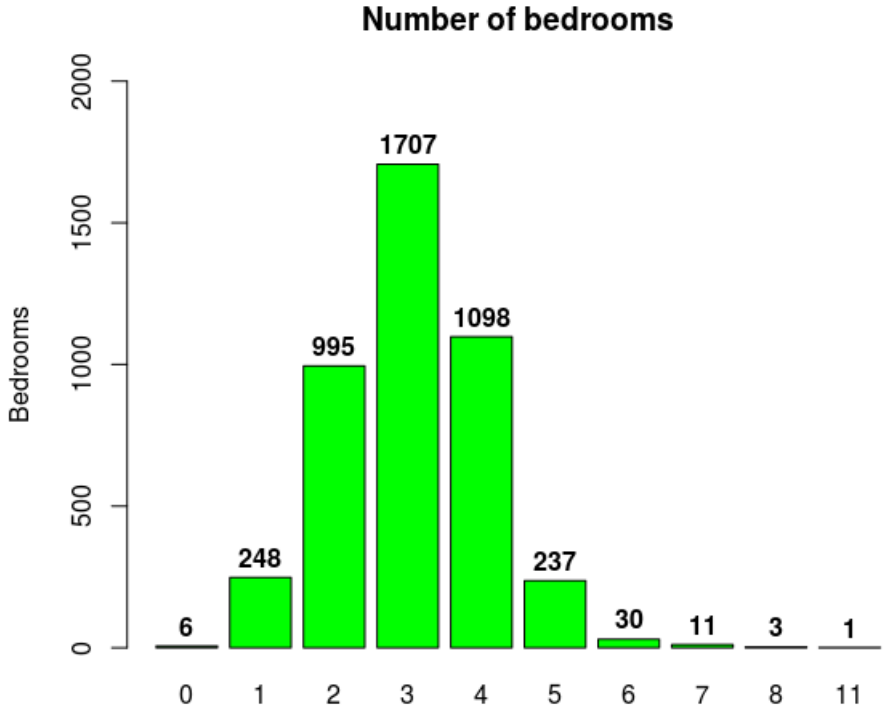

Рисунок 2.4 – Гистограмма спален

На рисунке 2.5 можно увидеть, что большинство квартир продаются в диапазоне цен от 250 до 750 тысяч долларов.

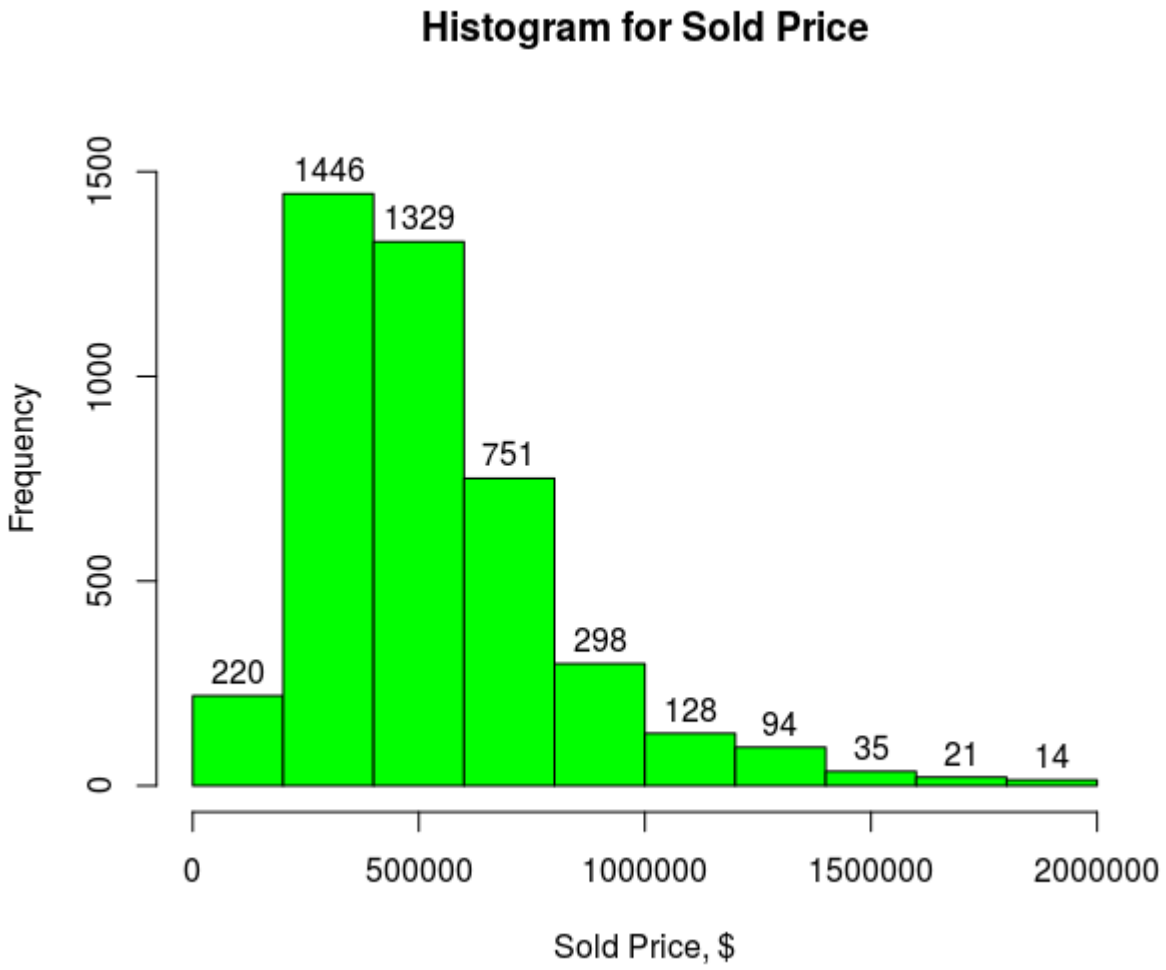

Рисунок 2.5 – Гистограмма стоимости продажи

2.2 Регрессионный анализ

В качестве первого способа анализа рынка недвижимости было решено использовать регресионный анализ.

В качестве инструмента для анализа данных было решено использовать язык программирования R. R — язык программирования для статистической обработки данных и работы с графикой, а также свободная программная среда вычислений с открытым исходным кодом в рамках проекта GNU. Язык создавался как аналогичный языку S, разработанному в Bell Labs, и является его альтернативной реализацией, хотя между языками есть существенные отличия. Изначально R был разработан сотрудниками статистического фа-

культета Оклендского университета Россом Айхэкой (англ. Ross Ihaka) и Робертом Джентлменом (англ. Robert Gentleman) [11].

У R есть ряд преимуществ по сравнению с Python. Он интуитивно понятен, а потому удобен, с точки зрения написания кода. Чтобы писать программы на R, необязательно соблюдать четкую структуру – можно просто вводить последовательный набор команд, и этого будет вполне достаточно.

Язык R создавался специально для анализа данных, поэтому все конструкции синтаксиса достаточно емки и понятны. Python — более универсальный и многоцелевой язык, что, естественно, усложняет его понимание.

Среди достоинств языка R можно отметить следующие:

– Удобные и понятные языковые конструкции

– Базовые статистические методы реализованы в качестве стандартных функций, что значительно повышает скорость разработки.

– Есть несколько отличных пакетов для визуализации. Можно строить и двумерную графику (диаграммы, боксплоты), а также и трехмерные модели. Результаты проведенной работы часто становятся значительно понятнее и выразительнее.

– Для R разработано огромное количество дополнительных пакетов.

Распределение цены объекта недвижимости в зависимости от его площади показана на рисунке 2.6. Можно заметить, что стоимость цены расчет с ростом площади, что ожидаемо, однако говорить о линейной зависимости не приходится. Коэффициент детерминации R^2 равен 0.15, что является достаточно низким показателем и не позволяет говорить о сильной взаимосвязи цены и площади.

Построим множественную модель регресии с учетом всех приведенных выше параметров. Анализ модели показан на рисунке.

Данные характеристики показывают нам, что такие параметры как общая площадь, количество гаражей, количество открытых парковочных мест, тип объекта недвижимости, количество душевых комнат и спален, средняя цены аренды недвижимости в районе являются значимыми и оказывают существенное влияние на формирование окончательной цены. Количество парковочных мест под навесом, средняя цены аренды недвижимости в районе, а также процент изменения арендной ставки не влияют на процесс формирования цены. Поэтому их можно исключить из модели.

Полученный коэффициент детерминации R^2 равен 0.48 оказался выше чем в модели парной регресии, однако он все еще недосточно велик чтобы точно формировать цену на объекты недвижимости. Далее модель стоимости строится отдельно по числу спален, так как этот параметр имеет наибольшее значение после площади объекта недвижимости и средней цены продажи

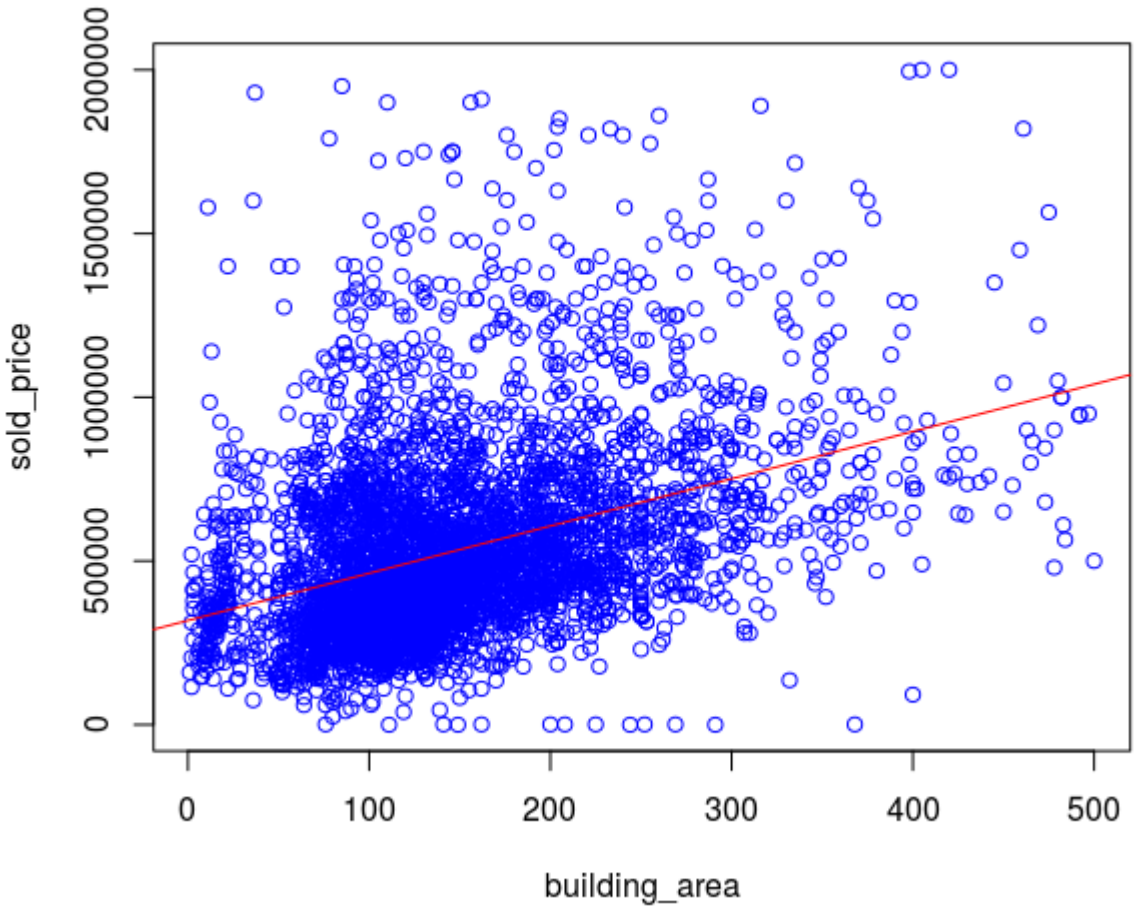

Рисунок 2.6 – Зависимость цены от площади

объектов недвижимости в данном районе и этот параметр является дискретным(рис. 2.7).

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       3.992e+04  9.679e+03   4.124 3.81e-05 ***
building_area     8.527e+02  5.762e+01  14.799  < 2e-16 ***
garage_count      1.146e+04  2.053e+03   5.584 2.55e-08 ***
open_spaces_count 1.035e+04  2.011e+03   5.148 2.79e-07 ***
property_type_id  4.493e+03  6.190e+02   7.258 4.88e-13 ***
bathrooms         3.669e+04  4.637e+03   7.912 3.44e-15 ***
bedrooms          2.821e+04  3.482e+03   8.102 7.57e-16 ***
average_price     1.789e-01  4.856e-03  36.845  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Рисунок 2.7 – Характеристики множественной модели регрессии

Построенные отдельно модели по числу спален показали схожие результаты и коэффициент детерминации R^2 равен 0.64. Полученная матрица корреляции представлена на рисунке 2.8. Исходя из данной матрицы можно сделать вывод что на конечную цену существенное влияние оказывает средняя цена проданной недвижимости, средняя цена аренды, тип объекта недвижимости и площадь объекта.

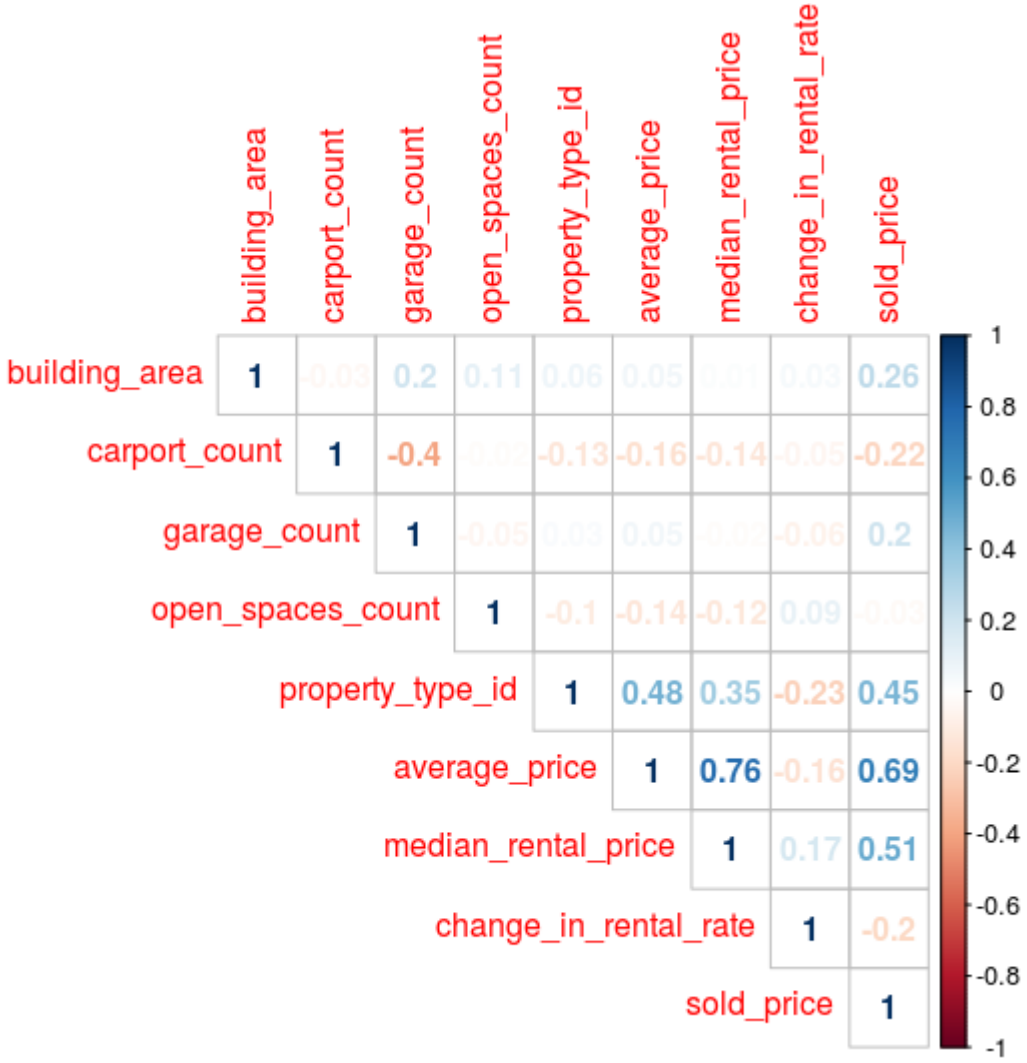

Рисунок 2.8 – Матрица корреляции

На рисунке 2.9 приведено сравнение прогнозируемой цены на недвижимость с фактической. Можно видеть, что полученная модель не позволяет точно прогнозировать цену на объект недвижимости ввиду большой погрешности. Исходя из этого можно сделать вывод, что предложенная модель оказалась неэффективной и не может быть использована для прогнозирования цены.

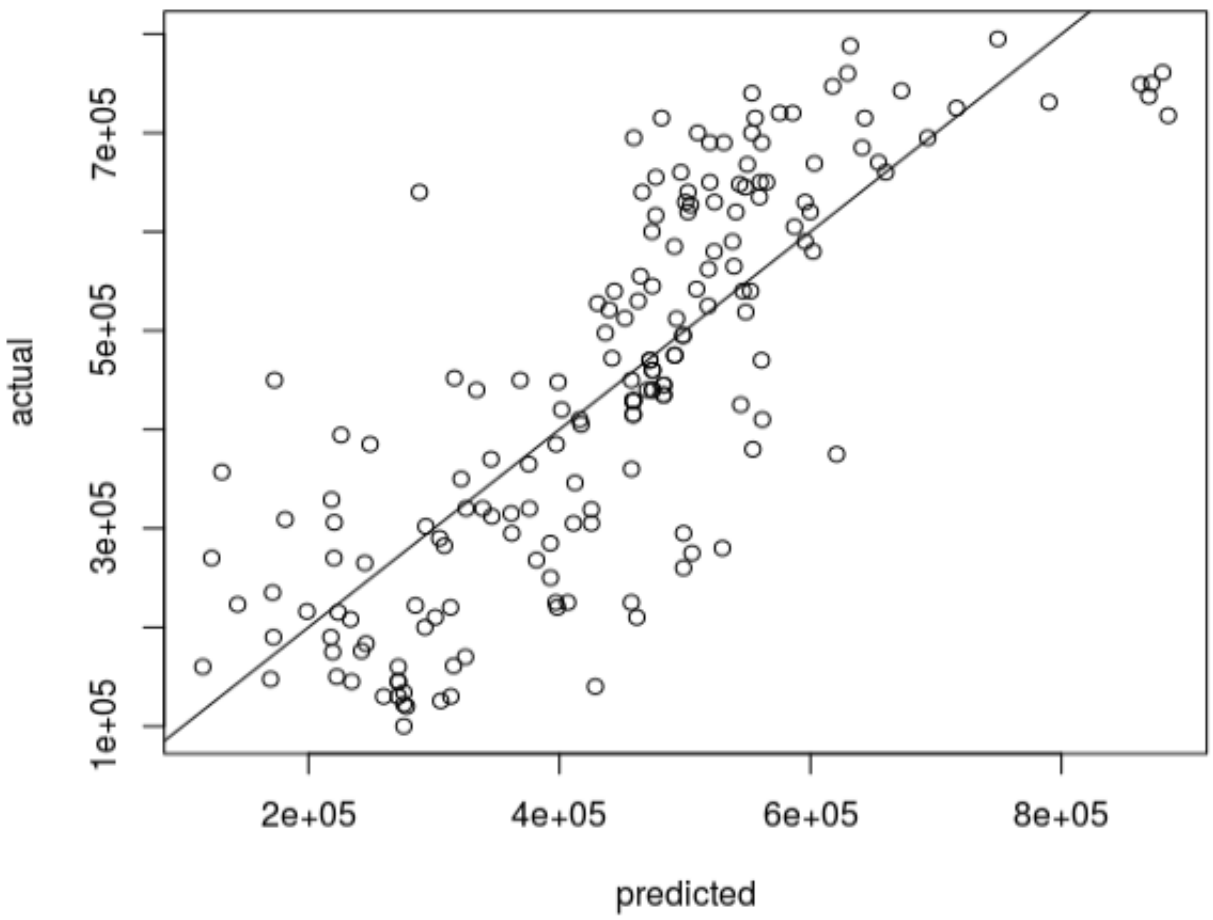

Рисунок 2.9 – Сравнение прогнозируемой цены с фактической

2.3 Анализ с использованием нейронных сетей

Для сравнения также было решено разработать методику оценки стоимости недвижимости с использованием нейронных сетей. Задача оценки недвижимости схематично представлена на рисунке 2.10

Для достижения цели необходимо выбрать факторы, влияющие на рыночную стоимость объектов недвижимости, подготовить выборку для обучения нейронной сети. Обучающая выборка построена для проектирования и обучения нейронной сети с учителем, поскольку такой тип нейронных сетей больше всего подходит для задач, когда имеется большой набор настоящих данных для обучения алгоритма. Исходя из сравнительного анализа нескольких типов нейронных сетей с учителем, проведенного в статье, было решено использовать нейронную сеть многослойный персептрон с использованием метода обратного распространения ошибки. Многослойным персептроном называют нейронную сеть прямого распространения, где входной сигнал распространяется от слоя к слою в прямом направлении. В общем представлении

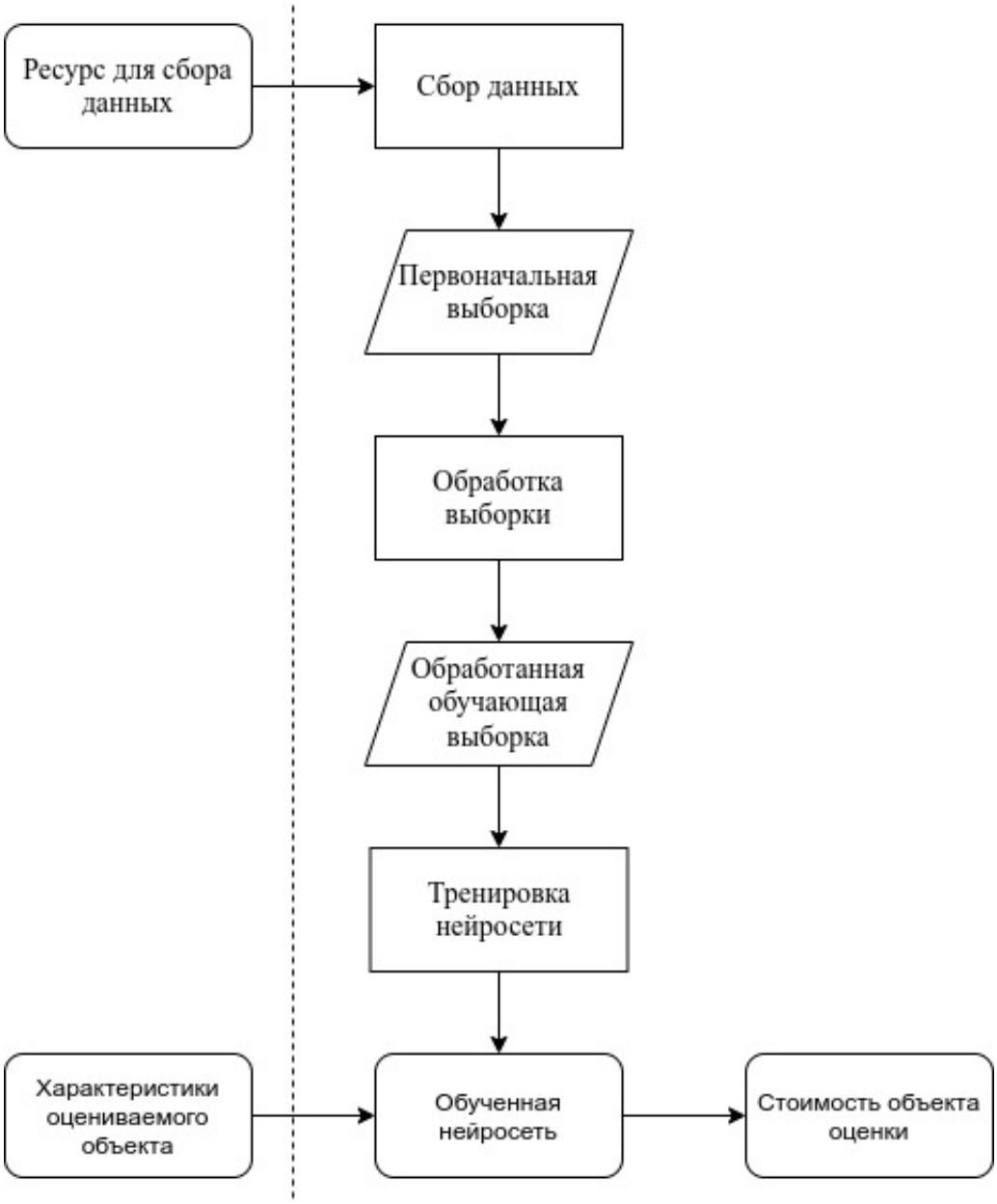

Рисунок 2.10 – Схема использования нейронных сетей для оценки стоимости недвижимости

такая нейронная сеть состоит из:

– множества входных узлов, образующих входной слой;
– одного или нескольких скрытых слоев вычислительных нейронов;
– одного выходного слоя нейронов.

Обощенная схема многослойного персептрона показана на рисунке 2.11

В качестве инструментального средства проектирования нейронной сети была выбрана STATISTICA Neural Networks. Для обучения многослойных персептронов в пакете STATISTICA Нейронные сети реализовано пять различных алгоритмов обучения. Это хорошо известный алгоритм обратного

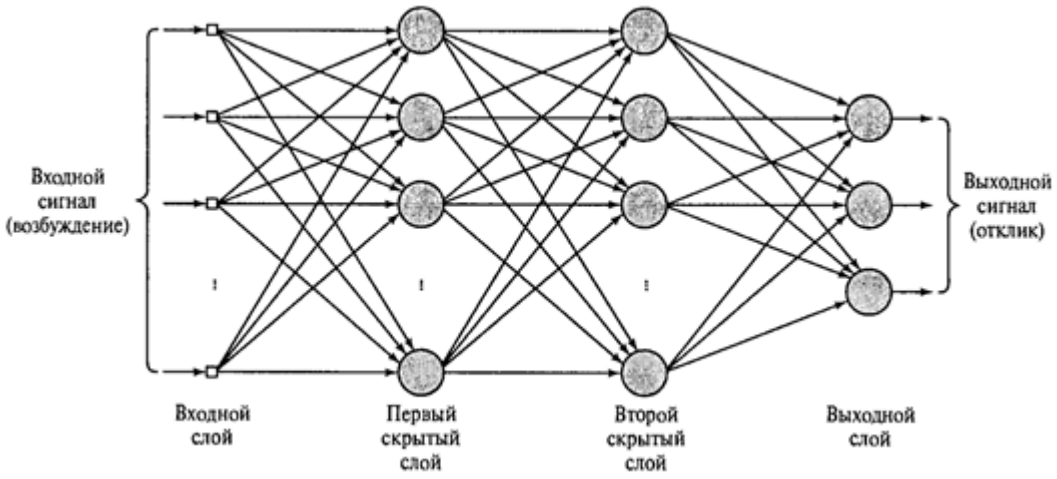

Рисунок 2.11 – Схема многослойного персептрона

распространения, быстрые методы второго порядка – спуск по сопряженным градиентам и Левенберга–Маркара, а также методы быстрого распространения и «дельта–дельта с чертой» (представляющие собой вариации метода обратного распространения, которые в некоторых случаях работают быстрее).

Алгоритм обратного распространения ошибки является популярным алгоритмом обучения нейронных сетей с учителем. В основе идеи алгоритма лежит использование выходной ошибки нейронной сети для вычисления величин коррекции весов нейронов в скрытых слоях

$$E = \frac{1}{2}\sum_{i=1}^{k}(y - y')^2 \tag{2.1}$$

где k — число выходных нейронов сети,
y — целевое значение,
y' — фактическое выходное значение.

Алгоритм является итеративным. На каждой итерации происходит прямой и обратный проходы. На прямом проходе входной вектор распространяется входов сети к ее выходам, в результате формируется выходной вектор, который соответствует фактическому состоянию весов. После вычисляется ошибка нейронной сети как разность между фактическим и целевым значениями. На обратном проходе эта ошибка распространяется от выхода сети к ее входам, и производится коррекция весов нейронов. Полученные данные дают возможность с достаточной точностью прогнозировать стоимость объектов недвижимости по заданным параметрам.

Реализованные алгоритмы обучения представлены на рисунке 2.12.

Кроме того, можно заметить, что в зависимости от алгоритма обучения и выбранной функции активации изменялась и конфигурация многослойного персептрона. Так, самой эффективной оказалась конфигурация с 21 нейроном на первом скрытом слое, на втором – 9. На обучение отводилось 4000 объектов из обучающей выборки, что составляет 70% от общего числа объектов, на контроль и тестирование по 15% - по 600 объектов. Обучение производится по методу обратного распространения ошибки.

Net. ID	Net. name	Training perf.	Test perf.	Validation p...	Algorithm
1	MLP 21-9-1	0,864290	0,848728	0,861748	BFGS 167
2	MLP 21-13-1	0,866216	0,843234	0,853091	BFGS 168
3	MLP 21-19-1	0,868197	0,846417	0,849407	BFGS 137
4	MLP 21-24-1	0,872672	0,853279	0,851128	BFGS 177
5	MLP 21-17-1	0,855069	0,834872	0,849904	BFGS 81
6	MLP 21-9-1	0,851975	0,830398	0,841425	BFGS 104

Рисунок 2.12 – Параметры качества реализованных алгоритмов обучения

Сравнение результатов ожидаемых с полученными представлено на рисунке 2.13. В первом столбце указана ожидаемая стоимость, во втором - полученная, в третьем - величина ошибки. Как можно заметить ошибка варьируется и может составлять как и небольшие значения так и значительные. Соотношение ожидаемых результатов с полученными показаны в виде графика на рисунке 2.14.

2.4 Сравнение результатов

На основании проведенных исследований можно утверждать, что применение нейронных сетей для прогнозирования стоимости объектов недвижимости более эффективно использования методов регрессионного анализа и может достаточно точно отражать рыночную стоимость недвижимости. Однако с увеличением числа выборки точность прогнозирования падает ввиду присутствия множества скрытых факторов(такие как время постройки дома, удаленность от различных сервисов, наличие или отсутствие мебели, качество постройки дома), не учитываемых в данном исследовании. Предложенные методы могут быть использованы продавцами для первичной оценки стоимости жилой недвижимости, а покупателями могут использоваться в качестве дополнительного источника информации, однако данная модель требует доработки.

Case name	Predictions spreadsheet for sold_price (teaching_properties) Samples: Train			
	sold_price Target	sold_price - Output 1. MLP 21-9-1	**sold_price - Residuals 1. MLP 21-9-1**	sold_price - Std. Res. 1. MLP 21-9-1
2733	510000,0	511840,3	-1840	-0,03265
1295	302000,0	303813,6	-1814	-0,03217
1320	367500,0	369257,4	-1757	-0,03118
2495	400000,0	401712,9	-1713	-0,03039
1592	680000,0	681602,2	-1602	-0,02842
996	580000,0	581535,9	-1536	-0,02725
3369	680000,0	681518,3	-1518	-0,02693
3371	680000,0	681518,3	-1518	-0,02693
3389	680000,0	681518,3	-1518	-0,02693
170	650000,0	651139,1	-1139	-0,02021
2491	390000,0	391096,5	-1096	-0,01945
671	300000,0	300925,0	-925	-0,01641
82	415000,0	415810,4	-810	-0,01438
1967	530000,0	530777,6	-778	-0,01379
943	538000,0	538762,4	-762	-0,01353
2268	606000,0	606731,8	-732	-0,01298
3338	310000,0	310687,6	-688	-0,01220
2387	535000,0	535623,0	-623	-0,01105
2660	205000,0	205532,0	-532	-0,00944
296	545000,0	545365,7	-366	-0,00649
2054	425000,0	425351,2	-351	-0,00623
94	407000,0	407141,6	-142	-0,00251
1596	440000,0	440137,6	-138	-0,00244
2696	336000,0	336088,4	-88	-0,00157
2457	550000,0	549981,6	18	0,00033
1971	428000,0	427846,9	153	0,00272
3555	285000,0	284539,4	461	0,00817
776	335000,0	334490,7	509	0,00903
1190	147500,0	146886,4	614	0,01089
2125	530000,0	529205,6	794	0,01409
964	510000,0	509183,3	817	0,01449
3708	465000,0	464082,5	918	0,01628
3163	450000,0	449040,9	959	0,01701
2729	480000,0	479004,4	996	0,01766
789	285000,0	283983,9	1016	0,01803
334	269500,0	268465,7	1034	0,01835
1159	245000,0	243940,2	1060	0,01880
3350	465000,0	463922,6	1077	0,01911
693	415000,0	413873,9	1126	0,01998
2991	620000,0	618607,4	1393	0,02470
841	450000,0	448514,1	1486	0,02636
764	355000,0	353506,3	1494	0,02650
1481	510000,0	508473,7	1526	0,02708
863	140000,0	138336,4	1664	0,02951
2676	525000,0	523324,9	1675	0,02972

Рисунок 2.13 – Сравнение полученных результатов с ожидаемыми

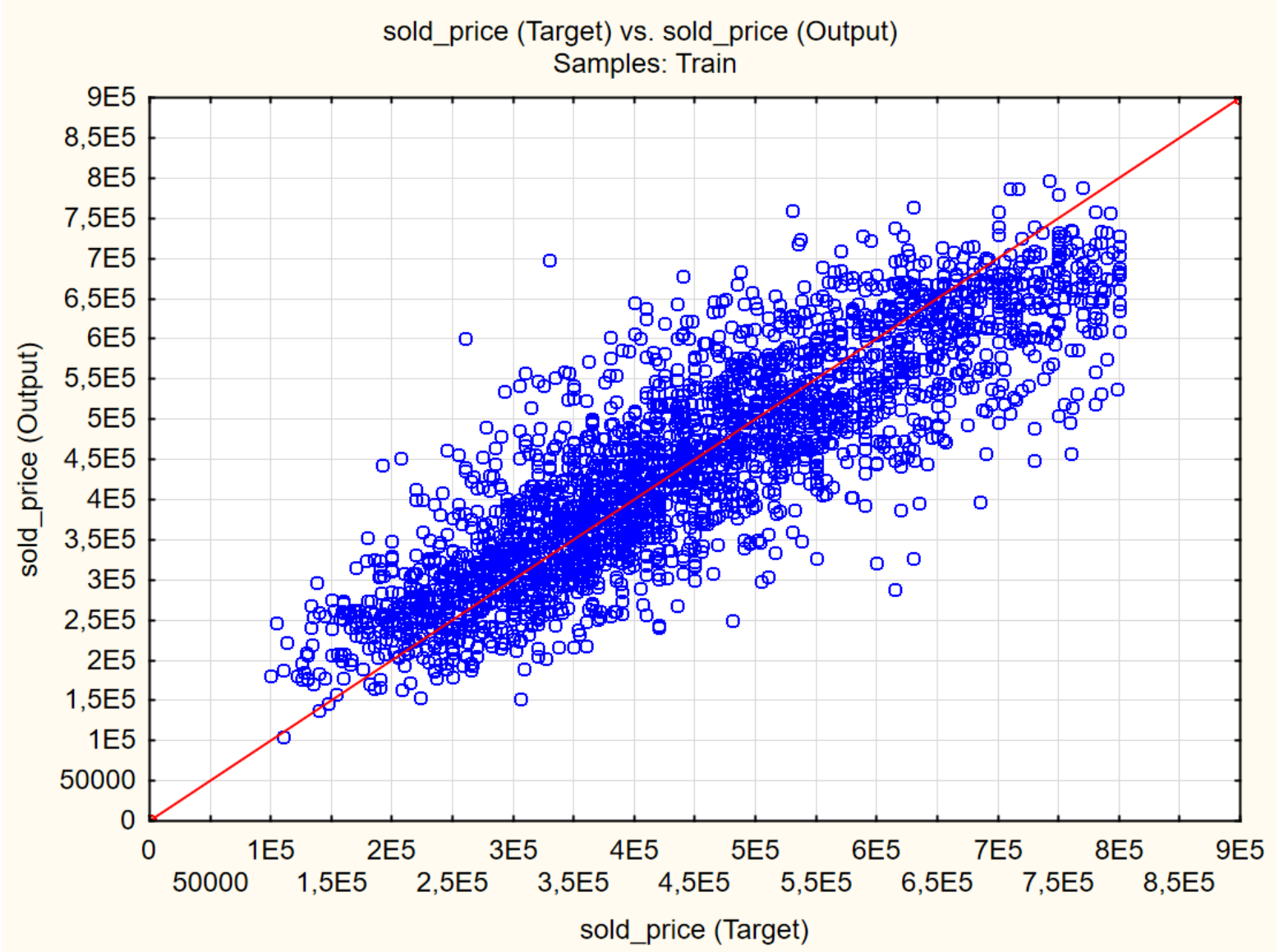

Рисунок 2.14 – Соотношение полученных результатов с ожидаемыми

ЗАКЛЮЧЕНИЕ

В ходе работы над магистерской диссертацией были изучены и проанализированы существующие способы и алгоритмы анализа данных. Были собраны данные об проданных объектах недвижимости на рынке за период с 2017 по 2020 год. Всего было собрано данных о более чем 5 тысяч проданных объектов недвижимости. После был выполнен регресионный анализ полученных данных и анализ с использованием нейронных сетей. Было проведено сравнение полученных результатов и сделаны соответствующие выводы об их эффективности и возможности реального использования.

На основании проведенных исследований можно утверждать, что применение нейронных сетей для прогнозирования стоимости объектов недвижимости более эффективно использования методов регрессионного анализа и может достаточно точно отражать рыночную стоимость недвижимости. Однако с увеличением числа выборки точность прогнозирования падает ввиду присутствия множества скрытых факторов(такие как время постройки дома, удаленность от различных сервисов, наличие или отсутствие мебели, качество постройки дома), не учитываемых в данном исследовании. Предложенные методы могут быть использованы продавцами для первичной оценки стоимости жилой недвижимости, а покупателями могут использоваться в качестве дополнительного источника информации, однако данная модель требует доработки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Специфика ценообразования на рынке жилья и факторы, влияющие на цену недвижимости [Электронный ресурс]. — Электронные данные. — Режим доступа: https://auditfin.com/fin/2009/2/Rodionova/Rodionova.pdf. — Дата доступа: 04.10.2020.

[2] Метод наименьших квадратов [Электронный ресурс]. — Электронные данные. — Режим доступа: https://wiki.loginom.ru/articles/least-squares-method.html. — Дата доступа: 04.10.2020.

[3] Коэффициент детерминации [Электронный ресурс]. — Электронные данные. — Режим доступа: https://wiki.loginom.ru/articles/coefficient-of-determination.html. — Дата доступа: 04.10.2020.

[4] Аппроксимация [Электронный ресурс]. — Электронные данные. — Режим доступа: https://wiki.loginom.ru/articles/approximation.html. — Дата доступа: 04.10.2020.

[5] Регрессионный анализ [Электронный ресурс]. — Электронные данные. — Режим доступа: https://wiki.loginom.ru/articles/regression-analysis.html. — Дата доступа: 04.10.2020.

[6] Карачун, С. В. Использование регрессионного анализа для оценки стоимости жилья / С. В. Карачун // Сборник работ 68-й научной научной конференции студентов и аспирантов БГУ, Минск. — 2011. — 5–8 Р.

[7] Нейорнные сети [Электронный ресурс]. — Электронные данные. — Режим доступа: https://habr.com/ru/post/312450/. — Дата доступа: 04.10.2020.

[8] Метод обратного распространения ошибки [Электронный ресурс]. — Электронные данные. — Режим доступа: https://otus.ru/nest/post/1592/. — Дата доступа: 04.10.2020.

[9] Алгоритм обучения RProp [Электронный ресурс]. — Электронные данные. — Режим доступа: https://basegroup.ru/community/articles/rprop. — Дата доступа: 04.10.2020.

[10] Е. А. Арефьева, Д . С. Костяев. Использование нейронных сетей для оценки рыночной стоимости недвижимости / Д . С. Костяев Е. А. Арефьева // Известия ТулГУ. — 2017. — no. 10. — 177–185 Р.

[11] Язык программирования R [Электронный ресурс]. — Электронные данные. — Режим доступа: https://www.r-project.org. — Дата доступа: 04.10.2020.

[12] Многослойный персептрон [Электронный ресурс]. — Электронные данные. — Режим доступа: http://www.aiportal.ru/articles/neuralnetworks/multi-perceptron.html. — Дата доступа: 04.10.2020.

Список публикаций соискателя

[1] Д. В. Голубко. Анализ цен на рынке недвижимости / Д. В. Голубко // IX Республиканская научно-практическая конференция «Вычислительные методы, модели и образовательные технологии»: сборник материалов. – Брест: БрГУ, 2020.

[2] Д. В. Голубко. Анализ цен на рынке недвижимости / Д. В. Голубко // Студенческий форум: электрон. научн. журн. 2021. № 2(138). URL: https://nauchforum.ru/journal/stud/138/84832.

ПРИЛОЖЕНИЕ А
Исходный код нейронной сети

```
using System;
public class Predict
{
   public static void __teaching_proper_MLP_21_9_1( double[] ContInputs, string[]
       CatInputs )
   {
     //"Input Variable" comment is added besides Input(Response) variables.
     int Cont_idx=0;
     int Cat_idx=0;
     double _building_area__ = ContInputs[Cont_idx++]; //Input Variable
     double _carport_count__ = ContInputs[Cont_idx++]; //Input Variable
     double _garage_count__ = ContInputs[Cont_idx++]; //Input Variable
     double _open_spaces_count__ = ContInputs[Cont_idx++]; //Input Variable
     double _bathrooms__ = ContInputs[Cont_idx++]; //Input Variable
     double _bedrooms__ = ContInputs[Cont_idx++]; //Input Variable
     double _number_of_sales__ = ContInputs[Cont_idx++]; //Input Variable
     double _average_price__ = ContInputs[Cont_idx++]; //Input Variable
     double _median_rental_price__ = ContInputs[Cont_idx++]; //Input Variable
     double _change_in_rental_rate__ = ContInputs[Cont_idx++]; //Input Variable
     string _property_type_id__ = CatInputs[Cat_idx++]; //Input Variable
    double[] __stat_max_input = new double[10];

    __stat_max_input[0]= 3.00000000000000e+002;
    __stat_max_input[1]= 8.00000000000000e+000;
    __stat_max_input[2]= 1.10000000000000e+001;
    __stat_max_input[3]= 1.00000000000000e+001;
    __stat_max_input[4]= 3.00000000000000e+000;
    __stat_max_input[5]= 5.00000000000000e+000;
    __stat_max_input[6]= 8.82000000000000e+002;
    __stat_max_input[7]= 9.75000000000000e+006;
    __stat_max_input[8]= 4.95000000000000e+005;
    __stat_max_input[9]= 2.89000000000000e+002;

    double[] __stat_min_input = new double[10];

    __stat_min_input[0]= 5.00000000000000e+001;
    __stat_min_input[1]= 0.00000000000000e+000;
    __stat_min_input[2]= 0.00000000000000e+000;
    __stat_min_input[3]= 0.00000000000000e+000;
    __stat_min_input[4]= 1.00000000000000e+000;
    __stat_min_input[5]= 1.00000000000000e+000;
    __stat_min_input[6]= 1.00000000000000e+000;
    __stat_min_input[7]= 1.00000000000000e+005;
    __stat_min_input[8]= 0.00000000000000e+000;
    __stat_min_input[9]= -1.00000000000000e+002;

    double[] __stat_max_target = new double[1];
    __stat_max_target[0]= 8.00000000000000e+005;

    double[] __stat_min_target = new double[1];
    __stat_min_target[0]= 1.00000000000000e+005;
```

```
double[,] __stat_i_h_wts = new double[9,21];
__stat_i_h_wts[0,0]=-3.74695424303563e+000;
__stat_i_h_wts[0,1]=8.80327911833777e+000;
__stat_i_h_wts[0,2]=-2.62200190596605e+000;
__stat_i_h_wts[0,3]=5.05430001768173e-001;
__stat_i_h_wts[0,4]=3.68607487480906e+000;
__stat_i_h_wts[0,5]=-1.75440268581406e+000;
__stat_i_h_wts[0,6]=-2.20945348821312e+000;
__stat_i_h_wts[0,7]=1.03998270578372e+001;
__stat_i_h_wts[0,8]=1.04827638844358e+000;
__stat_i_h_wts[0,9]=2.51551111309269e+001;
__stat_i_h_wts[0,10]=6.13405676927739e+000;
__stat_i_h_wts[0,11]=3.30868549492969e-001;
__stat_i_h_wts[0,12]=-1.12616483010226e+001;
__stat_i_h_wts[0,13]=5.17746414145129e-001;
__stat_i_h_wts[0,14]=1.75847380815526e+000;
__stat_i_h_wts[0,15]=-1.11294438117668e-002;
__stat_i_h_wts[0,16]=-5.47534203194262e+000;
__stat_i_h_wts[0,17]=-2.11268889244173e+000;
__stat_i_h_wts[0,18]=-1.67965020071394e+000;
__stat_i_h_wts[0,19]=6.58981526005262e+000;
__stat_i_h_wts[0,20]=-1.56351356505692e+000;
__stat_i_h_wts[1,0]=-1.67218776981920e-001;
__stat_i_h_wts[1,1]=-4.54625895453751e+000;
__stat_i_h_wts[1,2]=-1.93101133004951e+000;
__stat_i_h_wts[1,3]=6.51593686017428e-001;
__stat_i_h_wts[1,4]=2.82675810913731e+000;
__stat_i_h_wts[1,5]=1.09714341054083e+000;
__stat_i_h_wts[1,6]=-3.70679990197027e+000;
__stat_i_h_wts[1,7]=-1.32145453740465e+001;
__stat_i_h_wts[1,8]=-7.64009520160373e-001;
__stat_i_h_wts[1,9]=5.93410887559982e+000;
__stat_i_h_wts[1,10]=-2.88209932322992e+000;
__stat_i_h_wts[1,11]=6.01654405175162e-001;
__stat_i_h_wts[1,12]=4.45863171736489e-001;
__stat_i_h_wts[1,13]=7.79382168698538e-001;
__stat_i_h_wts[1,14]=6.92390455791189e+000;
__stat_i_h_wts[1,15]=-2.55659786958079e-002;
__stat_i_h_wts[1,16]=-1.31167861243867e+000;
__stat_i_h_wts[1,17]=-4.34668697752020e+000;
__stat_i_h_wts[1,18]=5.54361078572833e+000;
__stat_i_h_wts[1,19]=-3.07666876364801e+000;
__stat_i_h_wts[1,20]=-2.76385920991654e-002;
__stat_i_h_wts[2,0]=1.62066151745600e+000;
__stat_i_h_wts[2,1]=3.58930562091560e+000;
__stat_i_h_wts[2,2]=1.65555113543822e+000;
__stat_i_h_wts[2,3]=-2.56962376911591e+000;
__stat_i_h_wts[2,4]=9.93521360857537e-001;
__stat_i_h_wts[2,5]=4.64136454282061e-001;
__stat_i_h_wts[2,6]=2.31157571192460e+000;
__stat_i_h_wts[2,7]=9.67521392123815e+000;
__stat_i_h_wts[2,8]=-1.29726479208203e-001;
__stat_i_h_wts[2,9]=-4.94169476963029e+000;
__stat_i_h_wts[2,10]=-7.06357949123667e-001;
__stat_i_h_wts[2,11]=-6.33272066583827e-001;
```

```
__stat_i_h_wts[2,12]=-3.31452477499755e+000;
__stat_i_h_wts[2,13]=-7.32359269906306e-001;
__stat_i_h_wts[2,14]=2.57285067836654e+000;
__stat_i_h_wts[2,15]=1.98349797937427e-002;
__stat_i_h_wts[2,16]=-2.38758529831885e+000;
__stat_i_h_wts[2,17]=7.47692739614543e-001;
__stat_i_h_wts[2,18]=9.59032715657885e-001;
__stat_i_h_wts[2,19]=6.06808845370792e-001;
__stat_i_h_wts[2,20]=3.93708365375553e-001;
__stat_i_h_wts[3,0]=-1.23436265872341e-002;
__stat_i_h_wts[3,1]=-2.57819228727838e+000;
__stat_i_h_wts[3,2]=-1.66423187916516e+000;
__stat_i_h_wts[3,3]=-2.34375536850610e-001;
__stat_i_h_wts[3,4]=-1.89873592947885e+000;
__stat_i_h_wts[3,5]=5.76873456948957e-001;
__stat_i_h_wts[3,6]=1.27442717117625e+000;
__stat_i_h_wts[3,7]=-2.13547037893658e+001;
__stat_i_h_wts[3,8]=6.89110867679751e-001;
__stat_i_h_wts[3,9]=-1.91784711888474e+001;
__stat_i_h_wts[3,10]=4.28468412721788e-001;
__stat_i_h_wts[3,11]=1.72051890279987e+000;
__stat_i_h_wts[3,12]=-6.08001643827328e+000;
__stat_i_h_wts[3,13]=2.07081062657528e+000;
__stat_i_h_wts[3,14]=2.96103080116795e+000;
__stat_i_h_wts[3,15]=-2.32251700011142e-002;
__stat_i_h_wts[3,16]=1.90749543034651e+000;
__stat_i_h_wts[3,17]=-1.35655619119593e+000;
__stat_i_h_wts[3,18]=4.19443546949561e+000;
__stat_i_h_wts[3,19]=6.08039494488224e-001;
__stat_i_h_wts[3,20]=-3.25778318473112e+000;
__stat_i_h_wts[4,0]=1.48194310729123e+000;
__stat_i_h_wts[4,1]=-7.87146955202567e-003;
__stat_i_h_wts[4,2]=6.14975818083775e-001;
__stat_i_h_wts[4,3]=-7.81997823046458e-001;
__stat_i_h_wts[4,4]=3.99793097218536e+000;
__stat_i_h_wts[4,5]=1.20983321082893e+000;
__stat_i_h_wts[4,6]=-2.89994158199280e+000;
__stat_i_h_wts[4,7]=5.15177296317512e+000;
__stat_i_h_wts[4,8]=1.22431477733533e+000;
__stat_i_h_wts[4,9]=-2.58950566130288e+000;
__stat_i_h_wts[4,10]=-3.59688899894696e+000;
__stat_i_h_wts[4,11]=-5.52225410794502e-001;
__stat_i_h_wts[4,12]=5.47112559879194e+000;
__stat_i_h_wts[4,13]=-6.28848043599090e-001;
__stat_i_h_wts[4,14]=1.89260946530234e+000;
__stat_i_h_wts[4,15]=-4.41562832259195e-002;
__stat_i_h_wts[4,16]=3.43533707234905e-001;
__stat_i_h_wts[4,17]=-3.62537025642970e+000;
__stat_i_h_wts[4,18]=4.94990765160825e+000;
__stat_i_h_wts[4,19]=-1.05674989921733e+000;
__stat_i_h_wts[4,20]=-7.31244036527337e-001;
__stat_i_h_wts[5,0]=4.31737885705331e-001;
__stat_i_h_wts[5,1]=3.48829342808259e+000;
__stat_i_h_wts[5,2]=1.72545851097024e+000;
__stat_i_h_wts[5,3]=1.82270830908914e-001;
```

```
__stat_i_h_wts[5,4]=-1.96004945908883e+000;
__stat_i_h_wts[5,5]=-9.40424188552551e-001;
__stat_i_h_wts[5,6]=2.42099459659405e+000;
__stat_i_h_wts[5,7]=2.28158029552458e+001;
__stat_i_h_wts[5,8]=1.43518086997106e-001;
__stat_i_h_wts[5,9]=-2.16302134896775e+000;
__stat_i_h_wts[5,10]=-1.39965288357421e+000;
__stat_i_h_wts[5,11]=2.64704097558557e-001;
__stat_i_h_wts[5,12]=-4.78194838439253e-001;
__stat_i_h_wts[5,13]=2.85065664861725e-001;
__stat_i_h_wts[5,14]=9.35721803848455e-001;
__stat_i_h_wts[5,15]=-4.69069022973074e-003;
__stat_i_h_wts[5,16]=1.54067213338924e-002;
__stat_i_h_wts[5,17]=-8.17733208197051e-001;
__stat_i_h_wts[5,18]=1.83662298341902e+000;
__stat_i_h_wts[5,19]=-1.66723808056920e+000;
__stat_i_h_wts[5,20]=1.35079455285017e+000;
__stat_i_h_wts[6,0]=6.52567088914616e+000;
__stat_i_h_wts[6,1]=9.15001121330651e+000;
__stat_i_h_wts[6,2]=2.63850084796676e+000;
__stat_i_h_wts[6,3]=1.25446599101826e+000;
__stat_i_h_wts[6,4]=-1.16681168804019e+000;
__stat_i_h_wts[6,5]=4.54574469124889e+000;
__stat_i_h_wts[6,6]=-1.69562115938328e+000;
__stat_i_h_wts[6,7]=-1.22236296955904e+001;
__stat_i_h_wts[6,8]=1.29904379415973e+000;
__stat_i_h_wts[6,9]=-1.04060974613317e+001;
__stat_i_h_wts[6,10]=-1.59293725079775e+000;
__stat_i_h_wts[6,11]=-2.24759054311037e-001;
__stat_i_h_wts[6,12]=-8.16747696484144e-001;
__stat_i_h_wts[6,13]=-2.68868259700078e-001;
__stat_i_h_wts[6,14]=8.40970443431828e+000;
__stat_i_h_wts[6,15]=-2.09864990300200e-002;
__stat_i_h_wts[6,16]=1.61129513030147e+000;
__stat_i_h_wts[6,17]=-4.42662605263154e+000;
__stat_i_h_wts[6,18]=1.97471202190974e-001;
__stat_i_h_wts[6,19]=1.92230928065842e+000;
__stat_i_h_wts[6,20]=-9.74852127249932e-001;
__stat_i_h_wts[7,0]=1.72772826239836e-001;
__stat_i_h_wts[7,1]=1.09359711516470e-001;
__stat_i_h_wts[7,2]=2.58066042105559e-001;
__stat_i_h_wts[7,3]=1.20584552989270e+000;
__stat_i_h_wts[7,4]=-9.09656742081708e-001;
__stat_i_h_wts[7,5]=-4.75243024849759e-001;
__stat_i_h_wts[7,6]=9.68451401748806e-001;
__stat_i_h_wts[7,7]=3.93253519525386e+001;
__stat_i_h_wts[7,8]=-7.91034372476010e-001;
__stat_i_h_wts[7,9]=9.55526129807022e-001;
__stat_i_h_wts[7,10]=2.59614881309548e-001;
__stat_i_h_wts[7,11]=-9.78649115363495e-001;
__stat_i_h_wts[7,12]=-1.75114216371910e+000;
__stat_i_h_wts[7,13]=-1.24360086870149e+000;
__stat_i_h_wts[7,14]=1.53011155963180e+000;
__stat_i_h_wts[7,15]=-1.70270767245000e-002;
__stat_i_h_wts[7,16]=1.27093732060041e+000;
```

```
__stat_i_h_wts[7,17]=-1.67364103713981e+000;
__stat_i_h_wts[7,18]=2.41871363611252e+000;
__stat_i_h_wts[7,19]=-1.23294817534658e+000;
__stat_i_h_wts[7,20]=-3.82116114361364e-001;
__stat_i_h_wts[8,0]=-1.47323920679395e+001;
__stat_i_h_wts[8,1]=-2.25196528383047e+000;
__stat_i_h_wts[8,2]=2.32202426821384e+000;
__stat_i_h_wts[8,3]=1.00230133442532e+001;
__stat_i_h_wts[8,4]=-2.17489020470617e+001;
__stat_i_h_wts[8,5]=-5.03267007060702e-002;
__stat_i_h_wts[8,6]=5.15203774380202e+000;
__stat_i_h_wts[8,7]=2.75063148956440e-001;
__stat_i_h_wts[8,8]=1.03342791532057e-001;
__stat_i_h_wts[8,9]=-2.98404881607040e+000;
__stat_i_h_wts[8,10]=6.80984540973189e+000;
__stat_i_h_wts[8,11]=-1.77105915116862e-001;
__stat_i_h_wts[8,12]=-6.40674614861564e+000;
__stat_i_h_wts[8,13]=-1.80909189965960e-001;
__stat_i_h_wts[8,14]=6.14685287060225e+000;
__stat_i_h_wts[8,15]=-7.60900876206616e-003;
__stat_i_h_wts[8,16]=-7.60690357488769e+000;
__stat_i_h_wts[8,17]=-3.38102271792773e+000;
__stat_i_h_wts[8,18]=1.24240253760321e+001;
__stat_i_h_wts[8,19]=-4.70975700306944e+000;
__stat_i_h_wts[8,20]=1.57631756936350e-001;
double[,] __stat_h_o_wts = new double[1,9];
__stat_h_o_wts[0,0]=-3.24922697078351e+000;
__stat_h_o_wts[0,1]=-3.91731143779142e+000;
__stat_h_o_wts[0,2]=2.22664542752356e+000;
__stat_h_o_wts[0,3]=-4.36294212320045e+000;
__stat_h_o_wts[0,4]=2.89345698859494e+000;
__stat_h_o_wts[0,5]=-6.16519617547346e+000;
__stat_h_o_wts[0,6]=1.47881423067393e+000;
__stat_h_o_wts[0,7]=5.57429212728659e+000;
__stat_h_o_wts[0,8]=1.08331992929280e+000;
double[] __stat_hidden_bias = new double[9];
__stat_hidden_bias[0]=-6.75791157270563e+000;
__stat_hidden_bias[1]=2.62473198534524e+000;
__stat_hidden_bias[2]=-2.39862049259755e+000;
__stat_hidden_bias[3]=3.32369814262277e+000;
__stat_hidden_bias[4]=2.43782003400699e+000;
__stat_hidden_bias[5]=2.88726468421307e-001;
__stat_hidden_bias[6]=3.92557254113131e+000;
__stat_hidden_bias[7]=-1.84797833589827e+000;
__stat_hidden_bias[8]=3.07373137122875e+000;
double[] __stat_output_bias = new double[1];
__stat_output_bias[0]=1.61900573864608e+000;
double[] __stat_inputs = new double[21];
double[] __stat_hidden = new double[9];
double[] __stat_outputs = new double[1];
__stat_outputs[0] = -1.0e+307;
__stat_inputs[0]=_building_area__;
__stat_inputs[1]=_carport_count__;
__stat_inputs[2]=_garage_count__;
__stat_inputs[3]=_open_spaces_count__;
```

```
__stat_inputs[4]=_bathrooms__;
__stat_inputs[5]=_bedrooms__;
__stat_inputs[6]=_number_of_sales__;
__stat_inputs[7]=_average_price__;
__stat_inputs[8]=_median_rental_price__;
__stat_inputs[9]=_change_in_rental_rate__;

if( _property_type_id__=="1")
{
 __stat_inputs[10]=1;
}
else
{
 __stat_inputs[10]=0;
}

if( _property_type_id__=="10")
{

 __stat_inputs[11]=1;

}
else
{
 __stat_inputs[11]=0;
}

if( _property_type_id__=="12")
{
 __stat_inputs[12]=1;
}
else
{
 __stat_inputs[12]=0;
}

if( _property_type_id__=="19")
{
 __stat_inputs[13]=1;
}
else
{
 __stat_inputs[13]=0;
}

if( _property_type_id__=="2")
{
 __stat_inputs[14]=1;
}
else
{
 __stat_inputs[14]=0;
}

if( _property_type_id__=="21")
```

```
{
 __stat_inputs[15]=1;
}
else
{
 __stat_inputs[15]=0;
}

if( _property_type_id__=="25")
{
 __stat_inputs[16]=1;
}
else
{
 __stat_inputs[16]=0;
}

if( _property_type_id__=="26")
{
 __stat_inputs[17]=1;
}
else
{
 __stat_inputs[17]=0;
}

if( _property_type_id__=="4")
{
 __stat_inputs[18]=1;
}
else
{
 __stat_inputs[18]=0;
}

if( _property_type_id__=="5")
{
 __stat_inputs[19]=1;
}
else
{
 __stat_inputs[19]=0;
}

if( _property_type_id__=="6")
{
 __stat_inputs[20]=1;
}
else
{
 __stat_inputs[20]=0;
}

double __stat_delta=0;
double __stat_maximum=1;
```

```
  double __stat_minimum=0;
  int __stat_ncont_inputs=10;

  /*scale continuous inputs*/

  for(int __stat_i=0;__stat_i < __stat_ncont_inputs;__stat_i++)
  {
   __stat_delta = (__stat_maximum-__stat_minimum)/(__stat_max_input[__stat_i]-
       __stat_min_input[__stat_i]);
   __stat_inputs[__stat_i] = __stat_minimum - __stat_delta*__stat_min_input[__stat_i]+
       __stat_delta*__stat_inputs[__stat_i];
  }

  int __stat_ninputs=21;
  int __stat_nhidden=9;

  /*Compute feed forward signals from Input layer to hidden layer*/

  for(int __stat_row=0;__stat_row < __stat_nhidden;__stat_row++)
  {
    __stat_hidden[__stat_row]=0.0;
    for(int __stat_col=0;__stat_col < __stat_ninputs;__stat_col++)
    {
     __stat_hidden[__stat_row]= __stat_hidden[__stat_row] + (__stat_i_h_wts[__stat_row,
         __stat_col]*__stat_inputs[__stat_col]);
    }
   __stat_hidden[__stat_row]=__stat_hidden[__stat_row]+__stat_hidden_bias[__stat_row];
  }

  for(int __stat_row=0;__stat_row < __stat_nhidden;__stat_row++)
  {
   if(__stat_hidden[__stat_row]>100.0)
   {
    __stat_hidden[__stat_row] = 1.0;
   }
   else
   {
    if(__stat_hidden[__stat_row]<-100.0)
    {
     __stat_hidden[__stat_row] = 0.0;
    }
    else
    {
    __stat_hidden[__stat_row] = 1.0/(1.0+Math.Exp(-__stat_hidden[__stat_row]));
    }
  }
}

  int __stat_noutputs=1;
  /*Compute feed forward signals from hidden layer to output layer*/
  for(int __stat_row2=0;__stat_row2 < __stat_noutputs;__stat_row2++)
  {
   __stat_outputs[__stat_row2]=0.0;
  for(int __stat_col2=0;__stat_col2 < __stat_nhidden;__stat_col2++)
    {
```

```
    __stat_outputs[__stat_row2]= __stat_outputs[__stat_row2] + (__stat_h_o_wts[
        __stat_row2,__stat_col2]*__stat_hidden[__stat_col2]);
   }
  __stat_outputs[__stat_row2]=__stat_outputs[__stat_row2]+__stat_output_bias[
      __stat_row2];
 }

 for(int __stat_row=0;__stat_row < __stat_noutputs;__stat_row++)
 {
  if(__stat_outputs[__stat_row]>100.0)
  {
   __stat_outputs[__stat_row] = 1.0;
  }
  else
  {
   if(__stat_outputs[__stat_row]<-100.0)
   {
    __stat_outputs[__stat_row] = 0.0;
   }
   else
   {
    __stat_outputs[__stat_row] = 1.0/(1.0+Math.Exp(-__stat_outputs[__stat_row]));
   }
  }
 }

 /*Unscale continuous targets*/
 __stat_delta=0;
 for(int __stat_i=0;__stat_i < __stat_noutputs;__stat_i++)
 {
  __stat_delta = (__stat_maximum-__stat_minimum)/(__stat_max_target[__stat_i]-
      __stat_min_target[__stat_i]);
  __stat_outputs[__stat_i] = (__stat_outputs[__stat_i] - __stat_minimum + __stat_delta
      *__stat_min_target[__stat_i])/__stat_delta;
 }
   for(int __stat_ii=0; __stat_ii < __stat_noutputs; __stat_ii++)
   {
     Console.WriteLine(" Prediction{0} = {1}", __stat_ii+1, __stat_outputs[__stat_ii]);
   }
}
public static void Main (string[] args) {
  int argID = 0;
  double[] ContInputs = new double[10];
  int contID = 0;
  string[] CatInputs = new string[1];
  int catID = 0;
  if (args.Length >= 11)
  {
    ContInputs[contID++] = Double.Parse(args[argID++]);
    ContInputs[contID++] = Double.Parse(args[argID++]);
    ContInputs[contID++] = Double.Parse(args[argID++]);
    ContInputs[contID++] = Double.Parse(args[argID++]);
    ContInputs[contID++] = Double.Parse(args[argID++]);
    ContInputs[contID++] = Double.Parse(args[argID++]);
    ContInputs[contID++] = Double.Parse(args[argID++]);
```

```
        ContInputs[contID++] = Double.Parse(args[argID++]);
        ContInputs[contID++] = Double.Parse(args[argID++]);
        ContInputs[contID++] = Double.Parse(args[argID++]);
        CatInputs[catID++] = args[argID++];
      }
      else
      {
        string Comment = "";
        string Comment1 = "
            **************************************************************************\n";
        Comment += Comment1;
        string Comment2 = "Please enter at least 11 command line parameters in the
            following order for \nthe program to Predict.\n";
        Comment += Comment2;
        Comment += Comment1;
        string Comment3 = "building_area Type - double (or) integer\n";
        Comment += Comment3;
        string Comment4 = "carport_count Type - double (or) integer\n";
        Comment += Comment4;
        string Comment5 = "garage_count Type - double (or) integer\n";
        Comment += Comment5;
        string Comment6 = "open_spaces_count Type - double (or) integer\n";
        Comment += Comment6;
        string Comment7 = "bathrooms Type - double (or) integer\n";
        Comment += Comment7;
        string Comment8 = "bedrooms Type - double (or) integer\n";
        Comment += Comment8;
        string Comment9 = "number_of_sales Type - double (or) integer\n";
        Comment += Comment9;
        string Comment10 = "average_price Type - double (or) integer\n";
        Comment += Comment10;
        string Comment11 = "median_rental_price Type - double (or) integer\n";
        Comment += Comment11;
        string Comment12 = "change_in_rental_rate Type - double (or) integer\n";
        Comment += Comment12;
        string Comment13 = "property_type_id Type - String (categories are { \"1\" \"10\"
            \"12\" \"19\" \"2\" \"21\" \"25\" \"26\" \"4\" \"5\" \"6\" } )\n";
        Comment += Comment13;
        Comment += Comment1;
        System.Console.WriteLine(Comment);
        System.Environment.Exit(1);
      }
      __teaching_proper_MLP_21_9_1( ContInputs, CatInputs );
    }
}
```