Исследование контролируемых алгоритмов решения задачи прогнозирования динамики розничной торговли

Ohrimuk Ekaterina S.
Faculty of Computer Science and Technologies
Saint Petersburg Electrotechnical University "LETI"
Saint Petersburg, Russia
cat.oxrimuck@yandex.ru

Natalya V. Razmochaeva
Faculty of Computer Science and Technologies
Saint Petersburg Electrotechnical University "LETI"
St. Petersburg, Russia
nvrazmochaeva@etu.ru

Аннотация—В настоящей работе были изучены различные методы машинного обучения для прогнозирования динамики розничной торговли. Основная цель данной статьи – составление списка требований к разработке собственного алгоритма (или системы алгоритмов), основанного на методах машинного обучения. Для достижения цели был поставлен ряд задач, таких как: обзор предметной области и существующих методов для прогнозирования, выделение критериев, по которым будет проводиться сравнение, определение недостатков и преимуществ каждого из методов согласно критериям, и формирование перечня требований для реализации собственного алгоритма. Рассмотрены основные подходы и практические примеры использования машинного обучения для решения задачи прогнозирования динамики розничной торговли. Выявлены преимущества использования различных методов машинного обучения, такие как вариативность параметров, доступность, точность и другие. Выделен ряд метрик качества, которые используются для оценки качества рассмотренных алгоритмов. Обучение моделей производилось на реальном наборе данных из задачи, данные для которой опубликованы на интернет-ресурсе Kaggle.com. Предварительная обработка данных состояла в замене всех пропущенных значений, и замене категориальных переменных. Выполнена визуализация данных, а именно - диаграммы рассеяния для наглядности полученных результатов работы каждого алгоритма. В ходе практического анализа были выявлены алгоритмы, которые показали лучшие результаты по выбранным критериям, а именно Random Forest, Decision Tree и XGBRegressor. Сформулирован перечень требований для реализации собственного алгоритма, который будет представлять собой систему из совокупности рассмотренных алгоритмов, показавших высокие результаты. Система должна иметь возможность регулировать параметры модели, обладать высокой точностью прогноза целевой переменной, а также иметь простой интерфейс.

Ключевые слова— машинное обучение; прогноз; розничная торговля; алгоритмы обучения с учителем.

I. Введение

Розничная торговля – одна из значительных для общества отраслей хозяйства. Эффективная деятельность предприятия розничной торговли зависит от многих обстоятельств, в том числе она связана с оценкой разумных количеств товаров, приобретаемых для последующей реализации, что невозможно без прогнозирования спроса на товары, без учета ожидаемой интенсивности продаж.

Существует множество методов прогнозирования динамики розничной торговли, в том числе и в области алгоритмов обучения с учителем. Каждый из них обладает своими преимуществами и недостатками и имеет конкретную область применения. Необходима методика, которая позволяла бы уверенно предсказывать ожидаемые объемы продаж при колебаниях динамики реализации товаров. Предметом исследования являются алгоритмы обучения с учителем для решения задачи прогнозирования динамики розничной торговли. Объект исследования – алгоритмы обучения с учителем, применяемые для анализа данных в розничной торговле.

Цель данной работы заключается в формировании набора требований для реализации алгоритма прогнозирования динамики продаж, путем сравнительного анализа существующих подходов.

Для достижения цели потребуется решить ряд задач:

- Проанализировать существующие методы прогнозирования динамики розничной торговли.
- Выделить критерии, по которым будет проводиться сравнение.
- Определить недостатки и преимущества каждого из методов.

II. Обзор аналогов

Так как настоящая работа связана с прогнозированием в розничной торговле, то поиск аналогов был проведен среди научных исследований в области машинного обучения (набора методов, благодаря которым компьютер может находить в массивах изначально неизвестные взаимосвязи и закономерности), применяемых к анализу данных розничной торговли.

После изучения предложенных алгоритмов из таблицы 1, была сформирована таблица 2, которая содержит сравнение алгоритмов, представленных в таблице 1 по следующим выбранным критериям:

Таблица 1 РАССМАТРИВАЕМЫЕ АНАЛОГИ И ИХ ОСНОВНЫЕ СОСТАВЛЯЮЩИЕ

Название статьи	Используемые алгоритмы	Используемые метрики качества
Sales Forecast in E-commerce using Convolutional Neural Network [1].	Convolutional Neural Network (CNN) , ARIMA, FE+GBRT, DNN, CNN+WD, CNN+WD+TL, Single-CNN	Mean Squared Error (MSE)
Machine-Learning Models for Sales Time Series Forecasting [2].	ExtraTree, Linear Model, ARIMA, RandomForest, Neural Network, Stacking	Mean absolute error (MAE)
Sales forecasting using WaveNet within the framework of the Kaggle competition [3].	Convolutional Neural Network	Normalized weighted root mean squared logarithmic error (NWRMSLE)
Sales Demand Forecast in E-commerce using a Long Short-Term Memory Neural Network Methodology [4].	Globally training a Long Short-Term Memory network (LSTM)	Mean absolute percentage error (mMAPE)
Прогнозирование показателей инвестиций на основе метода деревьев решений [5]	Деревья решений, использующие алгоритм CART	MSE

- Вариативность параметров

Один и тот же алгоритм, применяемый для решения задачи прогнозирования динамики розничной торговли может показывать разную эффективность на различных данных, поэтому важно иметь возможность регулировать параметры модели для повышения точности прогноза применяемого алгоритма.

- Точность

Важно, чтобы полученные значения прогноза целевой переменной были близки к истинным значения целевой переменной, так как от этого напрямую зависит эффективность алгоритма.

Точности работы ARIMA, CNN были оценены в [1], где для оценки использовалась метрика среднего квадрата отклонения. Точности работы ExtraTree, Linear Model, RandomForest, Neural Network были оценены в [2], где для оценки использовалась средняя абсолютная ошибка. Точность работы LSTM была оценена в [4], где для оценки использовалась средняя абсолютная ошибка в процентах. Точность работы Decision Trees была оценена в [5], где для оценки использовалась метрика среднего квадрата отклонения.

- Доступность в реализации

Так как нашими моделями будут пользоваться обычные люди, не имеющие глубокий познаний в области машинного обучения, им важен простой интерфейс алгоритмов, важно использовать те алгоритмы, для которых реализация является очень удобной и простой. Обозначения, используемые для оценки доступности: 1 - на Python реализуется при помощи 1-2 библиотек, 2 - более 3 библиотек и разлизных методов.

Таблица 2 СРАВНЕНИЯ ПО КРИТЕРИЯМ

Алгоритмы	**Критерии**		
	Вариативность параметров	***Точность, %***	***Доступность в реализации***
CNN	+	>80	2
ARIMA	-	<60	1
ExtraTree	-	>80	1
Linear Model	-	[60-80]	1
LSTM	+	[60-80]	2
RandomForest	+	>80	1
Neural Network	-	>80	2
Decision Trees	+	>80	1

На основе обзора и анализа аналогов, были выбраны алгоритмы, которые показывают лучшие результаты по выбранным критериям и хорошие результаты для использования их в системах алгоритмов, согласно табл.2: Decision Tree, Random Forest и Linear Regression.

Большой набор рассматриваемых методов можно объяснить тем, что результатом исследования может являться не один метод, а сочетания разных методов.

III. ОПИСАНИЕ АЛГОРИТМОВ

- *Linear Regression*

Линейная регрессия на Python 3.6 реализуется с помощью sklearn.linear_model.LinearRegression (версия v0.21.3) [6].

С математической точки зрения линейный алгоритм в задачах регрессии выглядит следующим образом:

$$a(x) = w_0 + \sum_{j=1}^{d} w_j x^j,$$

где w_0 — свободный коэффициент,x^j— признаки, а w^j — их веса.

Если добавить (d + 1)-й признак, который на каждом объекте принимает значение 1, линейный алгоритм можно будет записать как:

$$a(x) = \sum_{j=1}^{d+1} w_j x^j = \langle w, x \rangle,$$

где используется обозначение $\langle w, x \rangle$ для скалярного произведения двух векторов [7].

- Decision Tree

Дерево решений на Python 3.6 реализуется с помощью sklearn.tree.DecisionTreeRegressor (версия v0.21.3) [8].

В общем, деревом решений можно назвать наглядную инструкцию, что делать в какой ситуации. В терминах машинного обучения можно сказать, что это модель, которая прогнозирует значение целевой переменной по нескольким атрибутам (признакам).

Структура дерева представляет собой «листья» и «ветки». На рёбрах («ветках») дерева решения записаны атрибуты, от которых зависит целевая функция, в «листьях» записаны значения целевой переменной, а в остальных узлах — атрибуты, по которым различаются случаи. Чтобы классифицировать новый случай, надо спуститься по дереву до листа и выдать соответствующее значение. Это жадный алгоритм - строится от корня к листьям [7].

- *Random Forest случайный лес*

Случайный лес на Python 3.6 реализуется с помощью sklearn.ensemble.RandomForestRegressor (версия v0.21.3) [9].

Случайный лес — это композиция деревьев, которые строятся независимо друг от друга. Композиция — это объединение N алгоритмов $b_1(x), \ldots, b_n(x)$в один. Общая идея заключается в том, чтобы обучить алгоритмы $b_1(x), \ldots, b_n(x)$, а затем усреднить полученные от них ответы:

$$a(x) = \frac{1}{N} \sum_{n=1}^{N} b_n(x),$$

где $a(x)$ - алгоритм, который возвращает среднее или знак среднего, который называется композицией N алгоритмов, $b_1(x), \ldots, b_n(x)$ – базовые алгоритмы [7].

- *XGBRegressor Градиентный бустинг*

Градиентный бустинг на Python 3.6 реализуется с помощью sklearn.ensemble.GradientBoostingClassifier (версия v0.21.3) [10].

Бустинг — это подход к построению композиций, в рамках которого:

- Базовые алгоритмы строятся последовательно, один за другим.
- Каждый следующий алгоритм строится таким образом, чтобы исправлять ошибки уже построенной композиции.

Благодаря тому, что построение композиций в бустинге является направленным, достаточно использовать простые базовые алгоритмы, например, неглубокие деревья.

В градиентном бустинге строящаяся композиция является суммой, а не усреднением базовых алгоритмов $b_1(x), \ldots, b_n(x)$:

$$a_N(x) = \sum_{n=1}^{N} b_n(x),$$

Описание алгоритма градиентного бустинга:

1. Инициализация: инициализация композиции $a_0(x) = b_0(x)$, то есть построение простого алгоритма b_0.
2. Шаг итерации:
   a) Вычисляется вектор сдвига:
   $s = -\blacktriangledown F = \left(\frac{-L'_z(y_1, a_{n-1}(x_1)), \ldots}{-L'_z(y_l, a_{n-1}(x_l))}\right)$.
   b) Строится алгоритм:
   $b_n = argmin_b \frac{1}{l} \sum_{i=1}^{l} (b(x_i) - s_i)^2$,
   параметры которого подбираются таким образом, что его значения на элементах обучающей выборки были как можно ближе к вычисленному вектору оптимального сдвига s.
   c) Алгоритм $b_n(x)$добавляется в композицию:
   $$a_n(x) = \sum_{m=1}^{n} b_m(x).$$
3. Если не выполнен критерий останова (критерий останова — это способ борьбы с переобучением), то выполнить еще один шаг итерации. Если критерий останова выполнен, остановить итерационный процесс [7].

- *KNN Regressor*

Метод ближайшего соседа на Python 3.6 реализуется с помощью sklearn.neighbors.KNeighborsRegressor (версия v0.21.3) [11].

Суть метода ближайшего соседа заключается в том, что новая точка относится к такому же классу, что и ближайшая к ней точка из обучающей выборки.

Взвешенный kNN для задачи регрессии определяется как:

$$a(x) = \frac{\sum_{i=1}^{k} w(x_{(i)}) x_{(i)}}{\sum_{i=1}^{k} w(x_{(i)})},$$

где x — новый объект, который требуется классифицировать, а $x_{(i)}$ — i-ый ближайший сосед из обучающей выборки [7].

IV. Error Measure

В качестве метрик для оценки прогнозирования используются:

- *Средняя абсолютная ошибка (MAE - Mean Absolute Error)*

Средняя абсолютная погрешность регрессии потерь на Python 3.6 реализуется с помощью sklearn.metrics.mean_absolute_error (версия v0.21.3) [12]

Средняя абсолютная ошибка определяется как:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} \quad |y_i - y\hat{}_i|,$$

где y_i - прогноз целевой переменной, $y\hat{}_i$- истинное значение целевой переменной, n – общее количество целевых переменных.

- *Средний квадрат отклонения (MSE – Mean Squared Error)*

Среднеквадратическая ошибка регрессии потерь на Python 3.6 реализуется с помощью sklearn.metrics.mean_squared_error (версия v0.21.3) [13]

Для вычисления среднеквадратической ошибки (MSE) все отдельные остатки регрессии возводятся в квадрат, суммируются, сумма делится на общее число ошибок:

$$MSE = \frac{\sum_{i=1}^{n} \quad (y_i - y\hat{}_i)^2}{n},$$

(см. пояснения к обозначениям этой формулы выше).

- *Корень из среднеквадратической ошибки (RMSE – Root Mean Squared Error)*

Квадратный корень из среднеквадратической ошибки регрессии потерь на Python 3.6 реализуется с помощью numpy.sqrt(sklearn.metrics.mean_squared_error())(версия v0.21.3) [13]. Квадратный корень из среднеквадратической ошибки вычисляется как:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} \quad (y_i - y\hat{}_i)^2}{n}},$$

(см. пояснения к обозначениям этой формулы выше).

Эти метрики качества позволяют объективно оценить точность моделей на тестовом наборе и сравнить используемые методы.

V. ЭКСПЕРИМЕНТЫ

Описание набора данных

Были протестированы выбранные алгоритмы на большом реальном наборе данных из задачи на Kaggle [14], которые делятся на наборы:

- stores.csv – содержит анонимную информацию о 45 магазинах, с указанием типа и размера магазина.
- train.csv – исторические данные обучения, которые охватывают период с 2010-02-05 по 2012-11-01. В этом файле содержатся следующие поля:

➢ Store - номер магазина.

➢ Dept - номер отдела.

➢ Date – неделя.

➢ Weekly_Sales - продажи для данного отдела в указанном магазине.

➢ IsHoliday – определяет, является ли неделя особой праздничной неделей.

- test.csv – идентичен train.csv, за исключением отсутствия целевой переменной - Weekly_Sales.
- feature.csv – содержит дополнительные данные, относящиеся к деятельности магазина, отдела и региона за указанные даты. Он содержит следующие поля:

➢ Store - номер магазина.

➢ Date – неделя.

➢ Temperature - средняя температура в регионе.

➢ Fuel_Price - стоимость топлива в регионе.

➢ MarkDown1-5 - анонимные данные, связанные с рекламными скидками, которые выполняет Walmart. Данные MarkDown доступны только после ноября 2011 года и доступны не для всех магазинов постоянно. Любое пропущенное значение помечается символом NA.

➢ CPI - индекс потребительских цен.

➢ Unemployment - уровень безработицы.

➢ IsHoliday - является ли неделя особой праздничной неделей.

Подготовка набора данных

Обучение моделей производится на 75% данных, тестирование – на 25%. Предварительно необходимо заменить на 0 пропущенные значения в признаках MarkDown1-5, преобразовать категориальную переменную 'Type' в числовую переменную для A = 1, B = 2, C = 3 и категориальную переменную IsHoliday в числовую переменную: если на этой неделе наступает дополнительный выходной, то 1 (= Да), иначе 2 (= Нет). Исходный код программ, которые готовились к измерениям, размещен в открытом доступе [15].

Оценка качества

Помимо рассмотренных алгоритмов из статей табл. 1 существуют еще и такие, как: XGBRegressor и KNN Regressor, которые также добавлены к сравнению.

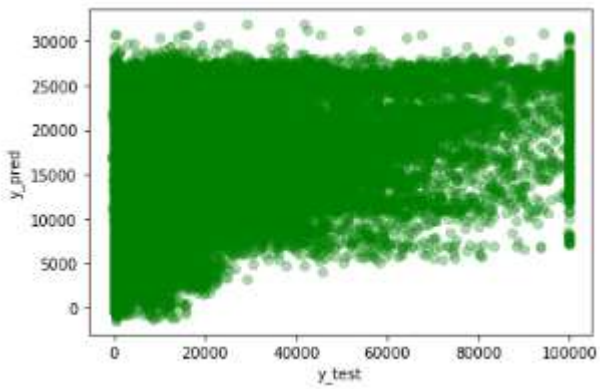

Рис. 1. Диаграмма рассеяния для Linear Regression

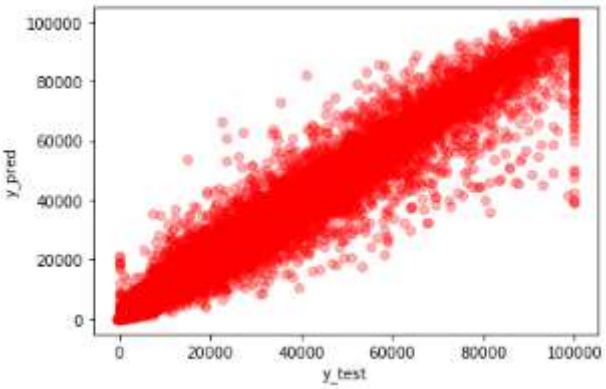

Рис. 2. Диаграмма рассеяния для Random Forest

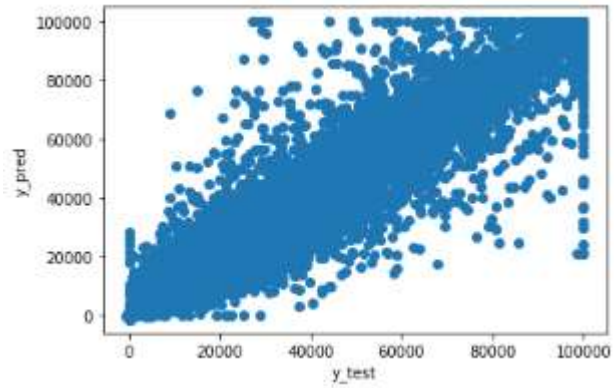

Рис. 3. Диаграмма рассеяния для Decision Tree

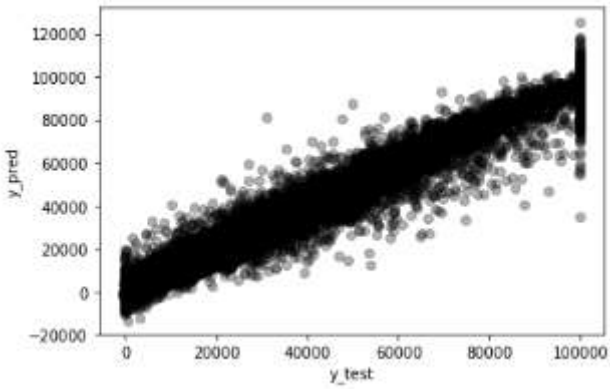

Рис. 4. Диаграмма рассеяния для XGBRegressor

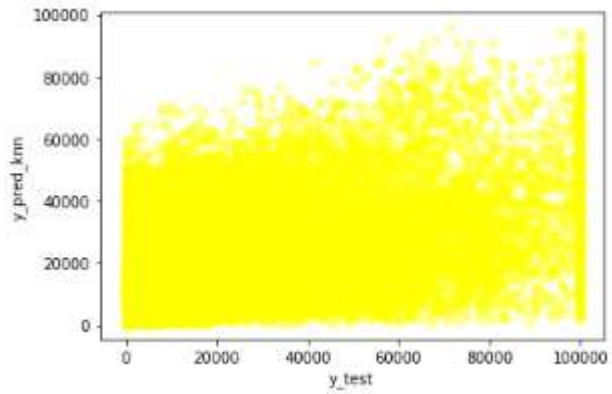

Рис. 5. Диаграмма рассеяния для KNN Regressor

На рис. 1-5 представлены диаграммы рассеивания, которые показывают значения, которые спрогнозировали наши модели и значения целевой переменной в виде точек на декартовой плоскости. По ним наглядно видно, что алгоритмы Linear Regression и KNN показали низкий уровень качества, в то время как XGBRegressor – очень высокий.

В табл. 3 представлены полученные результаты оценок качества для выбранных метрик после обучения моделей и применения их к тестируемой выборке.

ТАБЛИЦА 3. РЕЗУЛЬТАТ ОЦЕНКИ КАЧЕСТВА ДЛЯ АЛГОРИТМОВ ПО РАЗНЫМ МЕТРИКАМ КАЧЕСТВА

Алгоритмы	**Метрики качества**		
	MAE	***MSE***	***RMSE***
Linear Regression	13911	380118945	19496
Random Forest	1429	9413595	3068
XGBRegressor	1879	17866064	4226
KNN Regressor	1972	10960605	3310
Decision Tree	1879	17866064	4226

Согласно табл. 3, алгоритм Random Forest показал наилучшую точность из рассмотренных алгоритмов, алгоритмы Decision Tree и XGBRegressor показали допустимую и довольно высокую точность соответственно. Linear Regression и KNN Regressor далеки от истинных значений целевой переменной.

VI. Выводы

По результатам сравнительного анализа алгоритмов обучения с учителем для решения задачи прогнозирования динамики розничной торговли наилучшим из всех рассматриваемых алгоритмов оказался алгоритм Random Forest, который показал наивысшую точность прогноза целевой переменной для всех рассматриваемых метрик качества на тестовом наборе. Алгоритмы Decision Tree и XGBRegressor показали высокую точность, в то время как алгоритмы Linear Regression и KNN Regressor имеют большую ошибку при исследовании их на практике. Это может быть связано с тем, что они хорошо работают лишь в системе с другими алгоритмами и для их использования требуются особые условия.

В планах будущих исследований реализация системы алгоритмов, которая способна будет прогнозировать целевую переменную точнее, чем рассмотренные выше алгоритмы по отдельности и соответствовать следующим критериям:

- должна иметь возможность регулировать параметры модели для повышения точности прогноза применяемого алгоритма;
- должна обладать высокой точностью прогноза целевой переменной (не менее 80%);
- должна иметь простой интерфейс для обычных пользователей, без глубоких познаний в области машинного обучения.

Литература

[1] Kui Zhao, Can Wang, "Sales Forecast in E-commerce using Convolutional Neural Network", 8 p., 2017. Available at: https://arxiv.org/pdf/1708.07946.pdf

[2] Bohdan M. Pavlyshenko, "Machine-Learning Models for Sales TimeSeries Forecasting, 11 p., 2018. Available at: https://ru.scribd.com/document/427249237/data-04-00015-v2-pdf. (accessed: 2019-11-28)

[3] Glib Kechyn, Lucius Yu, Yangguang Zang, Svyatoslav Kechyn, “Sales forecasting using WaveNet within the framework of the Kaggle competition,” 6 p., 2018. Available at: https://arxiv.org/pdf/1803.04037.pdf. (accessed: 2019-11-28)

[4] Kasun Bandara, Peibei Shi, Christoph Bergmeir, Hansika Hewamalage, Quoc Tran, Brian Seaman, “Sales Demand Forecast in E-commerce using a Long Short-Term Memory Neural Network Methodology”, 16 p., 2019. Available at: https://arxiv.org/pdf/1901.04028.pdf. (accessed: 2019-11-28)

[5] Китова О.В., Колмаков И.Б., Пеньков И.А., "Прогнозирование показателей инвестиций на основе метода деревьев решений (in Russian)," Вестник РЭУ им. Г.В Плеханова №6 (90), 10 p., 2016. Available at: https://cyberleninka.ru/article/n/prognozirovanie-pokazateley-investitsiy-na-osnove-metoda-dereviev-resheniy/viewer

[6] class sklearn.linear_model.LinearRegression. Available at:https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html (accessed: 2019-11-28)

[7] Московский физико-технический институт, Яндекс, “Обучение на размеченных данных”. Available at: https://www.coursera.org/learn/supervised-learning/home/welcome.

[8] class sklearn.tree.DecisionTreeRegressor. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html. (accessed: 2019-11-28)

[9] class sklearn.ensemble.RandomForestRegressor. Available at https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html (accessed: 2019-11-28)

[10] class sklearn.ensemble.GradientBoostingClassifier. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html (accessed: 2019-11-28)

[11] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html. (accessed: 2019-11-28)

[12] sklearn.metrics.mean_absolute_error. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html (accessed: 2019-11-28)

[13] sklearn.metrics.mean_squared_error. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html (accessed: 2019-11-28)

[14] Walmart Recruiting - Store Sales Forecasting. Use historical markdown data to predict store sales. Available at: https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data (accessed: 2019-11-28)

[15] Available at: https://github.com/OhrimukKat/study_of_algorithms/blob/master/algorithms.ipynb(accessed: 2019-02-01)