Санкт-Петербургский политехнический университет Петра Великого

Институт машиностроения, материалов и транспорта

[**Высшая школа автоматизации и робототехники**](https://www.spbstu.ru/structure/vysshaya_shkola_avtomatizatsii_i_robototekhniki/)

**ЛАБОРАТОРНАЯ РАБОТА №2**

**«Методы классификации»**

по дисциплине

«Математические методы интеллектуальных технологий»

Выполнил

студент гр.3331506/60401 <*подпись*> Комаров А.Е.

Руководитель

<*подпись*> Орлова С.Р

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г.

Санкт-Петербург

2020

***Цель работы:*** освоение методов решения задачи классификации

***Задачи работы:***

* освоить основные алгоритмы классификации,
* научиться формировать признаки для использования в различных алгоритмах,
* научиться применять полученные знания об алгоритмах и навыки по формированию признаков для получения максимально качественных моделей.

# Ход работы и результаты

# Часть 1. Анализ и подготовка данных

1. Описание базы

В таблице 1 представлено описание базы *House Prices - Advanced Regression Techniques*.

Таблица 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Признак | Описание признака | Тип | Крайние значения | |
| Id | Номер записи | int64 | 1 | 1460 |
| SalePrice | Цена продажи объекта в долларах | int64 | 34900 | 755000 |
| MSSubClass | Тип жилья | int64 | 20 | 190 |
| MSZoning | Класс продажи | object |  |  |
| LotFrontage | Длина в футах улицы, соединенной с объетом | float64 | 21 | 313 |
| LotArea | Размер в квадратных футах | int64 | 1300 | 215245 |
| Street | Тип подъездной работы | object |  |  |
| Alley | Тип подъездной аллеи к объекту | object |  |  |
| LotShape | Общая форма объета | object |  |  |
| LandContour | Плокскостность объета | object |  |  |
| Utilities | Имеющиеся коммунальные услуги (газ, вода, электричество) | object |  |  |
| LotConfig | Конфигурация объета | object |  |  |
| LandSlope | Склон объета | object |  |  |
| Neighborhood | Район в пределах города Эймс | object |  |  |
| Condition1 | Близость к главной дорге или железной дороге | object |  |  |
| Condition2 | Близость к главной дорге или железной дороге (при наличии второй) | object |  |  |
| BldgType | Тип жилья | object |  |  |
| HouseStyle | Стиль жилья | object |  |  |
| OverallQual | Общее качество материалов и отделки | int64 | 1 | 10 |
| OverallCond | Общая оценка состояния | int64 | 1 | 9 |
| YearBuilt | Дата постройки | int64 | 1872 | 2010 |
| YearRemodAdd | Дата реконструкции | int64 | 1950 | 2010 |
| RoofStyle | Тип крыши | object |  |  |
| RoofMatl | Материал крыши | object |  |  |
| Exterior1st | Внешнее покрытие дома | object |  |  |
| Exterior2nd | Наружное покрытие дома (если используется более одного материала) | object |  |  |
| MasVnrType | Тип облицовки кладки | object |  |  |
| MasVnrArea | Площадь облицовки кладки в квадратных футах | float64 | 0 | 1600 |
| ExterQual | Качество наружных материалов | object |  |  |
| ExterCond | Текущее состояние материала снаружи | object |  |  |
| Foundation | Тип фундамента | object |  |  |
| BsmtQual | Высота подвала | object |  |  |
| BsmtCond | Общее состояние подвала | object |  |  |
| BsmtExposure | Подвальные стены на уровне сада или выхода | object |  |  |
| BsmtFinType1 | Качество готовой площади подвала | object |  |  |
| BsmtFinSF1 | Готовые квадратные футы типа 1 | int64 | 0 | 5644 |
| BsmtFinType2 | Качество второй законченной области подвала (если есть) | object |  |  |
| BsmtFinSF2 | Готовые квадратные футы типа 2 | int64 | 0 | 1474 |
| BsmtUnfSF | Незавершенные квадратные футы подвала | int64 | 0 | 2336 |
| TotalBsmtSF | Общая площадь подвала в квадратных футах | int64 | 0 | 6110 |
| Heating | Тип отопления | object |  |  |
| HeatingQC | Качество и состояние отопления | object |  |  |
| CentralAir | Наличие центральное кондиционирование | object |  |  |
| Electrical | Описание электрической системы | object |  |  |
| 1stFlrSF | Квадратные футы на первом этаже | int64 | 334 | 4692 |
| 2ndFlrSF | Квадратные футы второго этажа | int64 | 0 | 2065 |
| LowQualFinSF | Квадратные футы с отделкой низкого качества (все этажи) | int64 | 0 | 572 |
| GrLivArea | Квадратные футы жилой площади над уровнем земли | int64 | 334 | 5642 |
| BsmtFullBath | Полные ванные комнаты в подвале | int64 | 0 | 3 |
| BsmtHalfBath | Полуподвальные ванные комнаты | int64 | 0 | 2 |
| FullBath | Полные ванные комнаты над цокольным этажом | int64 | 0 | 3 |
| HalfBath | Полуванны над цокольным этажом | int64 | 0 | 2 |
| Bedroom | Количество спален над цокольным этажом | int64 | 0 | 8 |
| Kitchen | Количество кухонь | int64 | 0 | 3 |
| KitchenQual | Качество кухни | object |  |  |
| TotRmsAbvGrd | Общее количество комнат над цокольным этажом (без ванных комнат) | int64 | 2 | 14 |
| Functional | Рейтинг функциональности дома | object |  |  |
| Fireplaces | Количество каминов | int64 | 0 | 3 |
| FireplaceQu | Качество камина | object |  |  |
| GarageType | Расположение гаража | object |  |  |
| GarageYrBlt | Год постройки гаража | float64 | 1900 | 2010 |
| GarageFinish | Внутренняя отделка гаража | object |  |  |
| GarageCars | Размер гаража по вместимости машин | int64 | 0 | 4 |
| GarageArea | Размер гаража в квадратных футах | int64 | 0 | 1418 |
| GarageQual | Качество гаража | object |  |  |
| GarageCond | Состояние гаража | object |  |  |
| PavedDrive | Подъездная дорога с твердым покрытием | object |  |  |
| WoodDeckSF | Площадь деревянного настила в квадратных футах | int64 | 0 | 857 |
| OpenPorchSF | Площадь открытой веранды в квадратных футах | int64 | 0 | 547 |
| EnclosedPorch | Площадь веранды в квадратных футах | int64 | 0 | 552 |
| 3SsnPorch | Площадь веранды на три сезона в квадратных футах | int64 | 0 | 508 |
| ScreenPorch | Площадь крыльца в квадратных футах | int64 | 0 | 480 |
| PoolArea | Площадь бассейна в квадратных футах | int64 | 0 | 738 |
| PoolQC | Качество бассейна | object |  |  |
| Fence | Качество забора | object |  |  |
| MiscFeature | Прочие функции, не включенные в другие категории (лифт, сарай, теннисный корт и др.) | object |  |  |
| MiscVal | Ценность дополнительных функций | int64 | 0 | 15500 |
| MoSold | Месяц продажи | int64 | 1 | 12 |
| YrSold | Год продажи | int64 | 2006 | 2010 |
| SaleType | Тип продажи | object |  |  |
| SaleCondition | Состояние продажи | object |  |  |

1. Устранение “технических” ошибок

Первым шагом были выявлены признаки, не содержащие в себе достаточное количество информации для последующего анализа (малое количество ненулевых строк), и удалены из дальнейшего рассмотрения.

Была получена таблица с количеством не-*NA/null* записей в каждом признаке, после чего обнаружено, что 6 признаков могут быть удалены из дальнейшего рассмотрения. Ниже представлены фрагмент кода и таблица 2 – фрагмент таблицы *count.xlsx*, отсортированной по количеству не-*NA/null* записей.

data\_frame.count().to\_excel('count.xlsx')

data\_frame = data\_frame.drop(['PoolQC', 'Alley', 'FireplaceQu', 'Fence', 'MiscFeature', 'LotFrontage'], axis=1)

Таблица 2

|  |  |
| --- | --- |
| Признак | count() |
| PoolQC | 7 |
| MiscFeature | 54 |
| Alley | 91 |
| Fence | 281 |
| FireplaceQu | 770 |
| LotFrontage | 1201 |
| GarageType | 1379 |
| GarageYrBlt | 1379 |
| GarageFinish | 1379 |
| … | |

1. Устранение “смысловых” ошибок

Для устранения выбросов в числовых признаках использовался следующий фрагмент кода.

data\_frame = data\_frame[(data\_frame[["GrLivArea"]] < 4000).all(axis=1)]  
data\_frame = data\_frame[(data\_frame[["GarageCars"]] < 4).all(axis=1)]  
data\_frame = data\_frame[(data\_frame[["OverallQual"]] > 2).all(axis=1)]  
data\_frame = data\_frame[(data\_frame[["1stFlrSF"]] < 2000).all(axis=1)]  
data\_frame = data\_frame[(data\_frame[["GrLivArea"]] < 2500).all(axis=1)]  
data\_frame = data\_frame[(data\_frame[["TotalBsmtSF"]] < 2000).all(axis=1)]  
data\_frame = data\_frame[(data\_frame[["TotalBsmtSF"]] > 450).all(axis=1)]  
data\_frame = data\_frame[(data\_frame[["GarageArea"]] < 1000).all(axis=1)]

После устранения выбросов было произведено удаление всех строк, содержащих *NA/null* записи, и обновление номеров строк, что отражает представленный ниже фрагмент кода.

data\_frame = data\_frame.dropna()  
data\_frame.reset\_index(drop=True)

Далее был произведен перевод категориальных признаков в числовые.

for feature in data\_frame.columns:  
 if data\_frame[feature].dtype == object:  
 data\_frame[feature] = pandas.Categorical(data\_frame[feature])  
 sex\_map\_train = dict(zip(data\_frame[feature].cat.codes, data\_frame[feature]))  
 data\_frame[feature] = data\_frame[feature].cat.codes

Из-за большого количества признаков в базе на рисунках 1-3 представлены гистограммы только тех признаков, которые были выбраны в дальнейшем для анализа.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
| Рисунок 1 | |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
| Рисунок 2 | |

|  |  |
| --- | --- |
|  |  |
|  | |
| Рисунок 3 | |

1. Определение целевой переменной

В качестве целевой переменной выбрана цена продажи объекта в долларах. Так как выбранный признак является числовым, на основании его было произведено выделение трех ценовых групп - классов. Ниже представлен фрагмент кода, реализующий разбиение на три класса.

labels = [1, 2, 3]  
price\_group = pandas.cut(data\_frame['SalePrice'],  
 bins=[data\_frame['SalePrice'].min(),  
 145000,  
 200000,  
 data\_frame['SalePrice'].max()],  
 labels=labels)  
data\_frame.loc[:, 'price\_group'] = numpy.array(price\_group)

На рисунке 4 представлена гистограмма распределения объектов по выделенным ценовым категориям, которая демонстрирует сбалансированность выделенных классов.



Рисунок 4

1. Определение итогового набора данных

Для определения итогового набора данных были построены матрицы корреляции признаков между собой. Вследствие большого количества признаков и следующей из этого сложности анализа корреляций признаков с целевой переменной на основании графиков, были найдены корреляции всех признаков с ценой продажи объекта с помощью фрагмента кода, представленного ниже.

corr = data\_frame[['SalePrice'] + data\_frame.columns.to\_list()].corr().iloc[0]  
corr.sort\_values().to\_excel('correlation.xlsx')

На основании полученных данных были выбраны для дальнейшего анализа признаки с большими отрицательными и положительными значениями корреляции. В таблице 3 представлены значения корреляции выбранных признаков (выделены жирным шрифтом).

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| Признак | Корреляция | Признак | Корреляция |
| **ExterQual** | -0,63888 | GarageYrBlt | 0,532007 |
| **BsmtQual** | -0,60824 | **FullBath** | 0,54562 |
| **KitchenQual** | -0,58622 | YearRemodAdd | 0,553244 |
| **GarageFinish** | -0,53602 | **1stFlrSF** | 0,571015 |
| **HeatingQC** | -0,43744 | **YearBuilt** | 0,588213 |
| **GarageType** | -0,4121 | **TotalBsmtSF** | 0,598793 |
| BsmtExposure | -0,28911 | **GarageArea** | 0,599214 |
| … | | **GarageCars** | 0,637872 |
| Fireplaces | 0,41244 | **GrLivArea** | 0,66375 |
| **TotRmsAbvGrd** | 0,438394 | **OverallQual** | 0,785075 |
| Foundation | 0,509838 | SalePrice | 1 |

1. Создание генератора разбиений с параметрами

kf = KFold(n\_splits=3, shuffle=True, random\_state=12)

# Часть 2. Классификация

Под задачей классификации принимаем определение ценовой категории на основе объявлений о продаже домов.

В таблице 4 представлены результаты работы классификации следующими методами:

* методом ближайших соседей (*KNeighborsClassifier*);
* наивным байесовским классификатором (*GaussianNB*);
* решающими деревьями (*DecisionTreeClassifier*);
* методом опорных векторов (*SVC*);
* алгоритмом случайный лес (*RandomForestClassifier*).

Алгоритм случайный лес заключается в использовании ансамбля решающих деревьев. Основная идея заключается в использовании большого ансамбля [решающих деревьев](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%D0%BF%D1%80%D0%B8%D0%BD%D1%8F%D1%82%D0%B8%D1%8F_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B9), каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат получается хорошим.

В таблице представлен результат работы метода опорных векторов только с ядром радиальной базисной функции (ядро RBF), так как в результате экспериментов не удалось добиться сопоставимого результата у линейного метода опорных векторов.

Таблица 4



Подбор параметров для различных методов классификации осуществлялся при помощи функции *RandomizedSearchCV()*. В качестве применяемых параметров были отобрано самые значимые параметры метода классификации. Пример кода подбора наилучшего набора параметров и результата представлен ниже. В дальнейшем были отобраны усредненные параметры универсальные для трех видов обучающих данных: оригинальных, стандартизованных, нормализованных.

grid\_param = {'n\_neighbors': list(range(2, 100)),  
 'algorithm': ['auto', 'ball\_tree', 'kd\_tree', 'brute'],  
 'p': [1, 2, 3]}  
grid\_search = RandomizedSearchCV(KNeighborsClassifier(), param\_distributions=grid\_param, n\_iter=30)  
grid\_search.fit(X\_train, y\_train)  
print('type = ', type\_data, ' methot = ', str(KNeighborsClassifier()))  
print(grid\_search.best\_params\_)  
print(grid\_search.best\_score\_)

#Output

type = original methot = KNeighborsClassifier()

{'p': 1, 'n\_neighbors': 8, 'algorithm': 'auto'}

0.7348935847913353

Далее была написана функция *apply\_clustering\_method()*, применяющая метод классификации и возвращающая значение доли верного предсказания классификатора. Ниже представлен вызов вышеупомянутой функции для пяти методов классификации с выбранными параметрами.

apply\_clustering\_method(KNeighborsClassifier(n\_neighbors=35, algorithm='brute', p=2),  
 X\_train, y\_train, X\_test, y\_test, result\_table,   
 time\_result\_table, 5\*ikf + num\_meth, type\_data)  
  
apply\_clustering\_method(GaussianNB(var\_smoothing=0.075),  
 X\_train, y\_train, X\_test, y\_test, result\_table,   
 time\_result\_table, 5\*ikf + num\_meth, type\_data)  
  
apply\_clustering\_method(DecisionTreeClassifier(criterion='gini',

min\_samples\_split=10,   
 max\_depth=20),  
 X\_train, y\_train, X\_test, y\_test, result\_table,   
 time\_result\_table, 5\*ikf + num\_meth, type\_data)  
  
apply\_clustering\_method(Pipeline([("scaller", StandardScaler()),   
 ("svm\_clf", SVC(kernel="rbf", gamma=3, C=10))]),  
 X\_train, y\_train, X\_test, y\_test, result\_table,   
 time\_result\_table, 5\*ikf + num\_meth, type\_data)  
  
apply\_clustering\_method(RandomForestClassifier(n\_estimators=40, criterion='gini',   
 min\_samples\_split=6, max\_depth=40),  
 X\_train, y\_train, X\_test, y\_test, result\_table,   
 time\_result\_table, 5\*ikf + num\_meth, type\_data)

На основании таблицы можно отметить, что в большинстве методов стандартизованные данные дают лучшие результаты по сравнению с необработанными и нормализованными. В особенности у метода ближайших соседей и наивного байесовского классификатора.

Метод обучения алгоритмом случайный лес дал наибольшую точность результатов (около 80%), однако в то же время затратил наибольшее время на обучение модели.

# Заключение

По итогу работы был выбрана база объявлений продажи домов. В результате предварительного анализа были выявлены признаки с большим кол-вом пропусков. Однако их удаление было не совсем верным решением, но понятно это стало уже при завершении работ. Все категориальные признаки были приведены к числовым. Далее на основе значений корреляции признаков с целевым признаком – ценой продажи, было отобрано пятнадцать признаков с наибольшим абсолютным показателем корреляции. После оценки гистограмм распределения некоторые из параметров были обработаны для удаления выбросов.

В дальнейшем были проведены тестовые обучения моделей классификатора для установления наиболее подходящего кол-ва категорий цен, которые должен был определять классификатор. Выяснилось, что лучше всего цены разбить на три категории. Также это обосновано относительно небольшим размером база.

Далее были реализованы функции *get\_analiz()* и *apply\_clustering\_method()*, в результате работы которых выводится таблица со значениями времени обучения метода классификации и его предсказания, а также процентной доли верных предсказаний на тестовой выборке. Исходя из результатов наилучшим образом предсказывает значения метод случайного леса – около 80%, наихудшим результатом обладает метод опорных векторов – около 50%. Остальные методы имеют точность предсказания около 70%. Также, можно заметить, что наивысший процент верных предсказаний происходит на стандартизованных данных, а наименьший – при нормализованных данных.