МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Мегафакультет трансляционных информационных технологий

Факультет цифровых трансформаций

ПРОГНОЗИРОВАНИЕ АТМОСФЕРНЫХ ОСАДКОВ В АВСТРАЛИИ

Курсовая работа по дисциплине  
«Технологии машинного обучения»

Выполнил студент группы j4101     / Ю.И.Борисова/

Руководитель д.т.н., доцент     / Е.В. Котельников /

Работа защищена с оценкой     \_\_\_\_.\_\_\_\_.2021 г.

Санкт-Петербург

2021

Содержание

[Введение 3](#_Toc68740351)

[1 Теоретический обзор предметной области 4](#_Toc68740352)

[1.1 Название подраздела 4](#_Toc68740353)

[1.1.1 Название пункта 4](#_Toc68740354)

[1.1.2 Название пункта 4](#_Toc68740355)

[1.2 Выводы по разделу 5](#_Toc68740356)

[2 Название второго раздела 6](#_Toc68740357)

[2.1 Название подраздела 6](#_Toc68740358)

[2.1.1 Название пункта 6](#_Toc68740359)

[2.2 Выводы по разделу 6](#_Toc68740360)

[Заключение 7](#_Toc68740361)

[Библиографический список 8](#_Toc68740362)

[Приложения 9](#_Toc68740363)

[Приложение А. Листинг программы 9](#_Toc68740364)

[Приложение Б. Перечень сокращений и обозначений 10](#_Toc68740365)

# Введение

Машинное обучение (Machine Learning, ML) – это раздел теории искусственного интеллекта, предметом которого является поиск методов решения задач путем обучения в процессе решения сходных задач. Для построения таких методов используются средства алгебры, математической статистики, дискретной математики, теории оптимизации, численных методов, и других разделов математики.

Данная область является крайне актуальной в силу все возрастающих в настоящее время темпов развития и усовершенствования техники, а именно персональных компьютеров, приводящих к росту вычислительных мощностей. Описанный факт обуславливает возможность к использованию и изучению все более сложных моделей и систем при сравнительно небольших материальных и в ременных затратах на исследования.

Таким образом, проблема исследования заключается в агрегировании и актуализации знаний по основным тематическим разделам современного машинного обучения.

Целью курсовой работы является сбор и анализ теоретической информации по тематике, а также решение практической отраслевой задачи, для применения и апробации полученных знаний и умений.

Для достижения цели необходимо решить следующие задачи.

1. Рассмотреть подходы к предобработке данных.
2. Произвести теоретический обзор методов, применяемых при машинном обучении, в также задач, ими решаемых.
3. Рассмотреть подходы к подбору гиперпараметров моделей.

Курсовая работа включает два раздела. Первый раздел является теоретическим и включает обзор и рассмотрение базовых понятий, используемых в области машинного обучения. В нем представлено обобщение накопленного материала, планируемого к использованию во втором разделе – практическом.

Во втором разделе представлено описание экспериментов, произведенных в результате работы над реальной задачей (являющейся кейсом с сервиса Kaggle). В данной части курсового проекта предлагается проследить создание модели машинного обучения, включающее все этапы от проектирования и тестирования подходов, до конечной реализации.

# Теоретический обзор предметной области

## Цикл машинного обучения

### Методы предобработки данных

К методам предобработки данных в данном случае относятся любые манипуляции с исходным набором значений, производимые с целью повышения качества моделирования. Основными, рассматриваемыми в данной работе являются:

1. заполнение пропусков в данных дискретных и непрерывных случайных величин;
2. повышение размерности исходного набора данных;
3. масштабирование исходных признаков.

Зачатую набор данных, предоставляемый в качестве обучающей выборки, является неполным, в нем присутствуют пропуски – отсутствующие значения для некоторых переменных в случайных местах. Есть много способов решения данной проблемы. Наиболее тривиальным и в то же время наименее продуктивным является полное исключение из набора данных признаков, включающих пропуски. Однако такой подход приводит к нецелесообразно большой потере данных, поэтому при наличии хотя бы 20% значимой информации прибегают именно к заполнению пропусков. Простейшими способами является заполнение наиболее часто встречающимся (модой) значением для категориальных признаков и заполнение средним значением для некатегориальных (количественных) признаков [1].

В качестве более сложных способов заполнения пропусков для непрерывных случайных величин могут быть применены методы интерполяции: линейной, полиномиальной и др. [2]

Для повышения размерности исходного набора данных может быть использовано полиномиальное преобразование признаков (для полиномов разных степеней), что в некоторых случаях позволяет уловить некоторые неочевидные зависимости между переменными и как результат привести к лучшему качеству моделирования.

Масштабирование исходных признаков является очень распространенным (если не обязательным) подходом к преобразованию величин, измеряемых на неунифицированных шкалах, но являющихся частью одного датасета. В силу различной размерности, такие переменные как правило являются взаимно несоотносимыми и как следствие не сравнимыми, именно поэтому производится преобразование в единой шкале. Наиболее популярными методами являются приведение к значениям нормального распределения (от -1 до 1) и приведение к диапазону от минимального до максимального значений.

### Методы машинного обучения

Алгоритмы машинного обучения можно описать как обучение целевой функции f, которая наилучшим образом соотносит входные переменные X и выходную переменную Y: Y = f(X). Далее представлены наиболее популярные модели машинного обучения, опробованные при экспериментах.

*Линейная регрессия* — пожалуй, один из наиболее известных и понятных алгоритмов в статистике и машинном обучении. Линейную регрессию можно представить в виде уравнения, которое описывает прямую, наиболее точно показывающую взаимосвязь между входными переменными X и выходными переменными Y. Для составления этого уравнения нужно найти определённые коэффициенты B для входных переменных. Зная X, мы должны найти Y, и цель линейной регрессии заключается в поиске значений коэффициентов B0 и B1.

Для оценки регрессионной модели используются различные методы вроде линейной алгебры или метода наименьших квадратов. Линейная регрессия существует уже более 200 лет, и за это время её успели тщательно изучить.

*Логистическая регрессия* — ещё один алгоритм, пришедший в машинное обучение прямиком из статистики. Её хорошо использовать для задач бинарной классификации (это задачи, в которых на выходе мы получаем один из двух классов).

Логистическая регрессия похожа на линейную тем, что в ней тоже требуется найти значения коэффициентов для входных переменных. Разница заключается в том, что выходное значение преобразуется с помощью нелинейной или логистической функции.

Благодаря тому, как обучается модель, предсказания логистической регрессии можно использовать для отображения вероятности принадлежности образца к классу 0 или 1. Это полезно в тех случаях, когда нужно иметь больше обоснований для прогнозирования.

Как и в случае с линейной регрессией, логистическая регрессия выполняет свою задачу лучше, если убрать лишние и похожие переменные. Модель логистической регрессии быстро обучается и хорошо подходит для задач бинарной классификации.

*Дерево решений* можно представить в виде двоичного дерева, знакомого многим по алгоритмам и структурам данных. Каждый узел представляет собой входную переменную и точку разделения для этой переменной (при условии, что переменная — число). Листовые узлы — это выходная переменная, которая используется для предсказания. Предсказания производятся путём прохода по дереву к листовому узлу и вывода значения класса на этом узле.

Деревья быстро обучаются и делают предсказания. Кроме того, они точны для широкого круга задач и не требуют особой подготовки данных.

*Наивный Байес* — простой, но удивительно эффективный алгоритм. Модель состоит из двух типов вероятностей, которые рассчитываются с помощью тренировочных данных:

1. Вероятность каждого класса.
2. Условная вероятность для каждого класса при каждом значении x.

После расчёта вероятностной модели её можно использовать для предсказания с новыми данными при помощи теоремы Байеса. Если у вас вещественные данные, то, предполагая нормальное распределение, рассчитать эти вероятности не составляет особой сложности.

Наивный Байес называется наивным, потому что алгоритм предполагает, что каждая входная переменная независимая. Это сильное предположение, которое не соответствует реальным данным. Тем не менее данный алгоритм весьма эффективен для целого ряда сложных задач вроде классификации спама или распознавания рукописных цифр.

*К-ближайших соседей* — очень простой и очень эффективный алгоритм. Модель KNN (K-nearest neighbors) представлена всем набором тренировочных данных. Довольно просто, не так ли?

Предсказание для новой точки делается путём поиска K ближайших соседей в наборе данных и суммирования выходной переменной для этих K экземпляров.

Вопрос лишь в том, как определить сходство между экземплярами данных. Если все признаки имеют один и тот же масштаб (например, сантиметры), то самый простой способ заключается в использовании евклидова расстояния — числа, которое можно рассчитать на основе различий с каждой входной переменной.

KNN может потребовать много памяти для хранения всех данных, но зато быстро сделает предсказание. Также обучающие данные можно обновлять, чтобы предсказания оставались точными с течением времени.

Идея ближайших соседей может плохо работать с многомерными данными (множество входных переменных), что негативно скажется на эффективности алгоритма при решении задачи. Это называется проклятием размерности. Иными словами, стоит использовать лишь наиболее важные для предсказания переменные.

*Метод опорных векторов*, вероятно, один из наиболее популярных и обсуждаемых алгоритмов машинного обучения.

Гиперплоскость — это линия, разделяющая пространство входных переменных. В методе опорных векторов гиперплоскость выбирается так, чтобы наилучшим образом разделять точки в плоскости входных переменных по их классу: 0 или 1. В двумерной плоскости это можно представить как линию, которая полностью разделяет точки всех классов. Во время обучения алгоритм ищет коэффициенты, которые помогают лучше разделять классы гиперплоскостью.

Расстояние между гиперплоскостью и ближайшими точками данных называется разницей. Лучшая или оптимальная гиперплоскость, разделяющая два класса, — это линия с наибольшей разницей. Только эти точки имеют значение при определении гиперплоскости и при построении классификатора. Эти точки называются опорными векторами. Для определения значений коэффициентов, максимизирующих разницу, используются специальные алгоритмы оптимизации.

Метод опорных векторов, наверное, один из самых эффективных классических классификаторов, на который определённо стоит обратить внимание.

*Случайный лес* — очень популярный и эффективный алгоритм машинного обучения. Это разновидность ансамблевого алгоритма, называемого бэггингом.

Бутстрэп является эффективным статистическим методом для оценки какой-либо величины вроде среднего значения. Вы берёте множество подвыборок из ваших данных, считаете среднее значение для каждой, а затем усредняете результаты для получения лучшей оценки действительного среднего значения.

В бэггинге используется тот же подход, но для оценки всех статистических моделей чаще всего используются деревья решений. Тренировочные данные разбиваются на множество выборок, для каждой из которой создаётся модель. Когда нужно сделать предсказание, то его делает каждая модель, а затем предсказания усредняются, чтобы дать лучшую оценку выходному значению.

В алгоритме случайного леса для всех выборок из тренировочных данных строятся деревья решений. При построении деревьев для создания каждого узла выбираются случайные признаки. В отдельности полученные модели не очень точны, но при их объединении качество предсказания значительно улучшается.

Если алгоритм с высокой дисперсией, например, деревья решений, показывает хороший результат на ваших данных, то этот результат зачастую можно улучшить, применив бэггинг.

*Бустинг* — это семейство ансамблевых алгоритмов, суть которых заключается в создании сильного классификатора на основе нескольких слабых. Для этого сначала создаётся одна модель, затем другая модель, которая пытается исправить ошибки в первой. Модели добавляются до тех пор, пока тренировочные данные не будут идеально предсказываться или пока не будет превышено максимальное количество моделей.

AdaBoost был первым действительно успешным алгоритмом бустинга, разработанным для бинарной классификации. Именно с него лучше всего начинать знакомство с бустингом. Современные методы вроде стохастического градиентного бустинга основываются на AdaBoost.

AdaBoost используют вместе с короткими деревьями решений. После создания первого дерева проверяется его эффективность на каждом тренировочном объекте, чтобы понять, сколько внимания должно уделить следующее дерево всем объектам. Тем данным, которые сложно предсказать, даётся больший вес, а тем, которые легко предсказать, — меньший. Модели создаются последовательно одна за другой, и каждая из них обновляет веса для следующего дерева. После построения всех деревьев делаются предсказания для новых данных, и эффективность каждого дерева определяется тем, насколько точным оно было на тренировочных данных.

Так как в этом алгоритме большое внимание уделяется исправлению ошибок моделей, важно, чтобы в данных отсутствовали аномалии.

### Методы подбора гиперпараметров модели

Модели машинного обучения состоят из двух разных типов параметров:

* гиперпараметры - это все параметры, которые могут быть произвольно установлены пользователем перед началом обучения (например, количество оценок в случайном лесу).
* Параметры модели вместо этого изучаются во время обучения модели (например, веса в нейронных сетях, линейная регрессия).

Параметры модели определяют, как использовать входные данные для получения желаемого результата и которые изучаются во время обучения. Вместо этого гиперпараметры определяют, как модель в первую очередь структурирована.

Существуют следующие подходы к оптимизации гиперпараметров:

1. Ручной поиск
2. Случайный поиск
3. Grid Search
4. Автоматическая настройка гиперпараметров (Байесовская оптимизация, генетические алгоритмы)

При использовании ручного поиска выбираются некоторые гиперпараметры на основе личного суждения и опыта. Затем мы обучается модель, оценивается ее точность и процесс запускается заново. Этот цикл повторяется до достижения удовлетворительной точности.

*При случайном поиске* создается сетка гиперпараметров и модель обучается и тестируется только на некоторой случайной комбинации этих гиперпараметров. Зачастую для валидации модели и поиска наилучших гиперпараметров используется перекрестная проверка при которой тренировочный набор разделяется на N различных разделов, чтобы убедиться, что модель не соответствует данным.

Одним из наиболее распространенных методов перекрестной проверки является K-Fold Validation. В K-Fold обучающий набор разделяется на N разделов, а затем модель итеративно обучается используя N-1 разделы, и тестируется с оставшимся разделом (на каждой итерации оставшийся раздел меняется). После обучения N раз модели результаты обучения, полученные на каждой итерации, усредняются чтобы получить общие результаты обучения.

Использование перекрестной проверки при реализации оптимизации гиперпараметров может быть очень важным. Таким образом, возможно избежать использования некоторых гиперпараметров, которые действительно хорошо работают с тренировочными данными, но не очень хорошо работают с тестовыми данными.

В *Grid Search* настраивается сетка гиперпараметров и модель обучается и тестируется на каждой из возможных комбинаций. В связи с таким «жадным» подходом время на поиск наилучшей комбинации гиперпараметров возрастает крайне сильно.

Поиск по сетке медленнее по сравнению со случайным поиском, но в целом он может быть более эффективным, поскольку может проходить через все пространство поиска. Вместо этого случайный поиск может быть быстрее, но может пропустить некоторые важные моменты в пространстве поиска.

При использовании *автоматической настройки гиперпараметров* используемые гиперпараметры модели идентифицируются с использованием таких методов, как: байесовская оптимизация, градиентный спуск и эволюционные алгоритмы [3].

### Методы оценки качества моделей

В задачах машинного обучения для оценки качества моделей и сравнения различных алгоритмов используются метрики. Перед переходом к самим метрикам необходимо ввести важную концепцию для описания этих метрик в терминах ошибок классификации (для задачи классификации)— confusion matrix (матрица ошибок).  
При наличии алгоритма, предсказывающего принадлежность каждого объекта одному из классов, матрица ошибок классификации будет выглядеть следующим образом (Таблица 1).

|  |  |  |
| --- | --- | --- |
|  | Предсказанный 1 | Предсказанный 0 |
| Истинный 1 | True Positive (TP) | False Positive (FP) |
| Истинный 0 | False Negative (FN) | True Negative (TN) |

Основной метрикой считается accuracy — доля правильных ответов алгоритма:

(1.1)

Именно она и будет использоваться в практическом разделе курсовой работы, так как поставлена задача классификации.

## Выводы по разделу

Таким образом, на основе представленной информации, собранной в процессе теоретического обзора, можно составить общее впечатление о многообразии методов и подходов к машинному обучению. Под каждую конкретную задачу при поставленной цели оптимизации можно подобрать как наиболее простую и легко интерпретируемую модель, а также способ предобработки данных, так и тяжеловесную вычислительно емкую модель, сработающую в виде своеобразного «черного ящика». Важным моментом является подбор наиболее удобного и подходящего по соотношению «сложность - качество» варианта модели.

# Реализация модели классификации

## Постановка и решение задачи

В рамках практической части курсового проекта для проведения экспериментов была поставлена следующая прикладная задача для реализации методами машинного обучения: прогнозирование атмосферных осадков в Австралии.

### Описание набора данных

В качестве исходных данных выступил набор, предлагаемый сервисом Kaggle содержащий следующие переменные (Таблица 2).

Таблица 2. Краткая структура используемого набора данных

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Тип | Описание |
| 1 | Date | качественный | Дата наблюдения |
| 2 | Location | качественный | Город наблюдения |
| 3 | MinTemp | количественный | Минимальная суточная температура |
| 4 | MaxTemp | количественный | Максимальная суточная температура |
| 5 | Rainfall | количественный | Количество осадков |
| 6 | Evaporation | количественный | Испарение |
| 7 | Sunshine | количественный | Солнечность |
| 8 | WindGustDir | качественный | Направление ветра |
| 9 | WindGustSpeed | количественный | Скорость ветра |
| 10 | WindDir9am | количественный | Направление ветра в 9 утра |
| 11 | WindDir3pm | количественный | Направление ветра в 3 дня |
| 12 | WindSpeed9am | количественный | Скорость ветра в 9 утра |
| 13 | WindSpeed3pm | количественный | Скорость ветра в 3 дня |
| 14 | Humidity9am | количественный | Влажность в 9 утра |
| 15 | Humidity3pm | количественный | Влажность в 3 дня |
| 16 | Pressure9am | количественный | Давление в 9 утра |
| 17 | Pressure3pm | количественный | Давление в 3 дня |
| 18 | Cloud9am | количественный | Облачность в 9 утра |
| 19 | Cloud3pm | количественный | Облачность в 3 дня |
| 20 | Temp9am | количественный | Температура в 9 утра |
| 21 | Temp3pm | количественный | Температура в 3 дня |
| 22 | RainToday | качественный | Наличие дождя сегодня |
| 23 | RainTomorrow | качественный | Наличие дождя завтра |

### Предобработка данных и уточнение задачи

Целевой переменной для данной модели является поле RainTomorrow - качественный признак, характеризующий наличие или отсутствие дождя на завтрашний день. Таким образом, можно уточнить задачу до построения модели бинарной классификации по целевой переменной, при которой классу 1 будет соответствовать наличие дождя, а классу 0 – отсутствие.

При первичном анализе набора данных было выяснено, что во многих переменных имеются пропуски. Построение модели на данных с пропусками является нежелательным, поэтому было принято решение заполнить пропуски простейшим возможным способом. Для категориальных признаков - заполнением значением моды, наиболее часто встречаемым значением, для количественных признаков – заполнением средним значением по выборке.

После устранения пропусков была произведена генерация новых признаков посредством разбиения показателя Date (даты измерения) на три отдельных показателя – число, месяц и год наблюдения.

Категориальный признак Location (город наблюдения) было решено исключить из предикторов, так как он обладает большим количеством уникальных значений, и без привязки к пространству (координатам) не будет информативным при обучении, но сделает вычисления более ресурсоемкими.

В рамках предподготовки данных была произведена бинарная кодировка всех категориальных признаков. Наибольшее количество классов наблюдалось у полей, отражающих направление ветра.

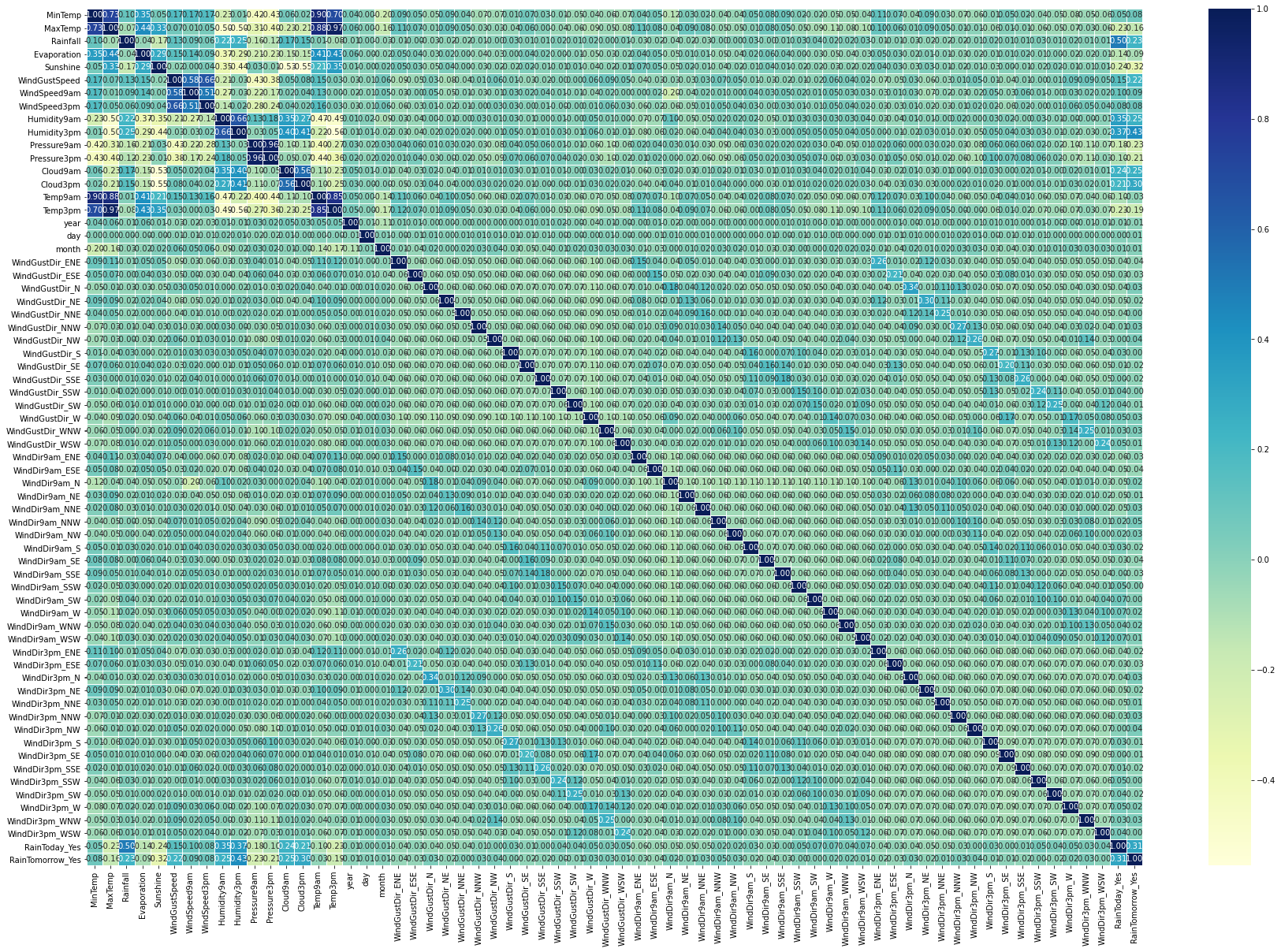
Для исключения малозначимых предикторов из рассмотрения была построена матрица попарных корреляций признаков (Рисунок 1).

Рисунок 1. Матрица попарных корреляций сгенерированных признаков (изображение в лучшем качестве см. в Приложении)

После анализа корреляционной матрицы, был сделан вывод о том, что целевая переменная имеет связь далеко не с каждым из сгенерированных предикторов и для дальнейшего моделирования было принято решение оставить только те переменные, корреляция которых с целевой переменной составляет более 0.2. Такими предикторами являются:

1. наличие дождя сегодня;
2. облачность в 3 и в 9 часов;
3. влажность в 3 и в 9 часов;
4. давление в 3 и в 9 часов;
5. количество осадков;
6. солнечность.

После отбора значимых предикторов было произведено масштабирование всех некатегориальных признаков в соответствии со шкалой от минимального до максимального дначений, что является необходимым при работе с имеющимся набором данных, так как для различных показателей используется различная размерность.

На данном этапе предподготовка данных была завершена, однако ее элементы будут присутствовать при дальнейшем подборе оптимальной структуры модели.

### Подбор оптимального алгоритма и гиперпараметров

В качестве дополнительного этапа предподготовки данных можно произвести повышение размерности исходного набора обучающих данных полиномиальным преобразованием, однако в данном случае выбор степени полинома не является тривиальной задачей, в связи с чем данный шаг было решено перенести в блок поиска оптимальной структуры модели.

Так как выбранный набор данных не предполагает наличие разбиения на тестовую и обучающую выборку разбиение было реализовано самостоятельно в отношении 80 на 20%, где 80% - обучающая выборка, 20% - тестовая выборка.

Для проведения экспериментов с подбором оптимального алгоритма и гиперпараметров для него были выбраны следующие алгоритмы:

1. Логистическая регрессия;
2. AdaBoost;
3. Градиентный бустинг;
4. Деревья решений.

Поиск наиболее успешной модели производится «жадным» поиском по сетке. Также в этот поиск был включен подбор оптимальной степени полинома как части предподготовки данных. Степень качества работы модели оценивалась по показателю accuracy, часто применимому к задачам классификации. В качестве метода валидации использовался алгоритм перекрестной проверки K-Fold Validation с количеством разбиений 10.

Таким образом в результате подбора параметров для каждой модели лучшими оказались гиперпараметры, приведенные в Таблице 2. Лучший показатель Accuracy показала модель на основе градиентного бустинга.

|  |  |  |  |
| --- | --- | --- | --- |
| Название алгоритма | Лучшие гиперпараметры | Степень полинома | Accuracy |
| Логистическая регрессия | {'linear\_model\_\_C': 1.1} | 1 | 0.7812 |
| AdaBoost | {'ensemble\_\_n\_estimators': 99} | 1 | 0.7795 |
| Градиентный бустинг | {'ensemble\_\_learning\_rate': 0.1, 'ensemble\_\_max\_depth': 1, 'ensemble\_\_n\_estimators': 95} | 1 | 0.7815 |
| Деревья решений |  |  |  |

## Выводы по разделу

Таким образом в данном разделе были применены на практике описанные в теоретической части подходы к машинному обучению, а именно:

1. Способы предобработки данных;
2. Некоторые из рассмотренных моделей машинного обучения;
3. Подбор оптимальных гиперпараметров для моделей;
4. Оценка результатов классификации.

В результате последовательного применения полученных знаний был построен полный цикл машинного обучения, завершающийся обучением модели и ее валидацией. Результаты работы различных алгоритмов представлены для сравнения в табличном виде, на основе чего можно сделать вывод о их пригодности к использованию и относительно высоком качестве.

# Заключение

Таким образом в результате работы были достигнуты поставленные задачи, а именно в рамках теоретического рахдела:

1. Рассмотрены подходы к предобработке данных;
2. Произведен теоретический обзор методов, применяемых при машинном обучении;
3. Рассмотрены подходы к подбору гиперпараметров моделей.

В рамках практического раздела теоретическая информация применена на практическом примере для реальной задачи. Была произведена предобработка данных описанными в теоретическом разделе методами, построены несколько моделей машинного обучения, решающие задачи классификации. Методом поиска по сетке подобраны оптимальные гиперпараметры для моделей, а также степени полинома для предобработки данных. Произведена валидация моделей и рассчитаны качества предсказания, на основе чего сделаны выводы о пригодности всех построенных моделей к решению поставленной задачи – прогнозированию осадков в Австралии.

# Библиографический список

1.К.A. Майков, П.А. Гаврилов, Обзор методов предобработки, используемых для решения задач классификации в условиях неполноты данных. (2016), 140–145

2. Е.С. Родчикова, Анализ применения способов заполнения пропусков в данных во временных рядах в экологических исследованиях ». (2012).

3. Timofeev, A. V. (2020). Method for hyperparameter tuning in machine learning tasks for stochastic objects classification. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, *5*(October), 667–676. https://doi.org/10.17586/2226-1494-2020-20-5-667-676

# Приложения

## Приложение А. Листинг программы

**Модуль чтения данных (read\_csv.py).**

**import** **csv**, **sys**

filename = 'some.csv'

# Стараемся хорошо комментировать код

**with** open(filename, newline='') **as** f:

reader = csv.reader(f)

**try**:

**for** row **in** reader:

print(row)

**except** csv.Error **as** e:

sys.exit('file *{}*, line *{}*: *{}*'.format(filename, reader.line\_num, e))

**Модуль записи данных (write\_csv.py).**

**import csv**

# Каждый фрагмент кода должен сопровождаться комментариями

**with** open('some.csv', 'w', newline='') **as** f:

writer = csv.writer(f)

writer.writerows(someiterable)

## Приложение Б. Перечень сокращений и обозначений

|  |  |
| --- | --- |
| ACL | – Association for Computational Linguistics (Ассоциация компьютерной лингвистики); |
| AJAX | – Asynchronous JavaScript And Xml (технология обращения к серверу без перезагрузки страницы); |
| API | – Application Programming Interface (интерфейс прикладного программирования). |