**1. Что такое Линейные Модели?**

**Общее представление**: Это класс моделей машинного обучения, которые очень широко используются и имеют долгую историю изучения.

**Принцип работы:** Они делают прогнозы, используя линейную функцию от входных признаков. Это означает, что прогноз получается путем взвешенного суммирования значений признаков плюс некоторое смещение.

**2. Линейные Модели для Регрессии**

**Задача регрессии**: Предсказать непрерывное числовое значение (например, цену дома, температуру).

Формула: Прогноз y^ (читается "y с крышкой") вычисляется так:

y^ = w[0] \* x[0] + w[1] \* x[1] + ... + w[p] \* x[p] + b

x[0], x[1], ..., x[p] — это значения признаков для одного конкретного объекта. p+1 - общее количество признаков.

w[0], w[1], ..., w[p] — это веса или коэффициенты модели. Каждый вес показывает, насколько сильно соответствующий признак влияет на прогноз. Эти веса модель изучает в процессе обучения.

b — это свободный член, сдвиг или константа (intercept). Это базовое значение прогноза, когда все признаки равны нулю, или смещение прогнозируемой линии/плоскости относительно начала координат. Этот параметр также изучается моделью.

**Ограничение**: Линейные модели делают довольно сильное предположение, что зависимость между признаками и целевой переменной является линейной. Это может быть слишком просто для сложных данных, где зависимость нелинейная.

**3. Обычный Метод Наименьших Квадратов (OLS) - LinearRegression**

**Цель**: Это самый базовый и классический способ найти параметры w и b для линейной регрессии.

**Принцип**: OLS ищет такие w и b, которые минимизируют среднеквадратичную ошибку (Mean Squared Error, MSE) на обучающем наборе данных.

**MSE**: Это среднее значение квадратов разностей между реальными значениями (y) и прогнозами модели (y^). То есть, модель старается сделать так, чтобы квадраты вертикальных расстояний от точек данных до предсказанной линии/плоскости были в сумме как можно меньше.

**Недостаток**: У OLS нет встроенного механизма для контроля сложности модели. Если признаков много (особенно если их больше, чем примеров данных), или если признаки сильно коррелируют между собой, OLS может привести к переобучению.

**4. Проблема Переобучения (Overfitting)**

**Что это**: Модель слишком хорошо "подстраивается" под особенности обучающих данных, включая случайный шум. В результате она отлично работает на обучающих данных, но плохо обобщает свои знания на новые, невиданные (тестовые) данные.

**Признак**: Высокое качество (например, высокий R²) на обучающем наборе и значительно более низкое качество на тестовом наборе.

**Почему возникает с OLS:** При большом количестве признаков модель становится очень гибкой и может найти очень большие по модулю коэффициенты w, чтобы точно подогнаться под обучающие точки. Эти большие коэффициенты делают модель очень чувствительной к малейшим изменениям во входных данных, что вредит обобщению.

**5. Регуляризация: Борьба с Переобучением**

**Идея**: Чтобы предотвратить переобучение, нужно ограничить сложность модели. Регуляризация — это способ добавить к основной цели (минимизации ошибки на обучении) дополнительное ограничение на параметры модели (w).

**Цель ограничения**: Заставить модель предпочитать более простые решения, обычно — решения с меньшими по модулю коэффициентами w. Это делает модель менее чувствительной к шуму в обучающих данных.

**6. Гребневая Регрессия (Ridge Regression) - L2 Регуляризация**

**Принцип**: Это линейная модель (y^ = wx + b), которая при поиске w и b минимизирует не только MSE, но и дополнительный штраф, пропорциональный сумме квадратов коэффициентов w (L2-норма).  
 Минимизировать (MSE + alpha \* Σ(w[i]^2))

Параметр alpha: Контролирует силу регуляризации.

alpha = 0: Нет регуляризации, Ridge превращается в обычный OLS.

alpha > 0: Штраф применяется. Чем больше alpha, тем сильнее штраф, тем сильнее коэффициенты w "притягиваются" к нулю (но не становятся точно равными нулю), и тем проще становится модель.

**Эффект**: Уменьшает величину коэффициентов, стабилизирует модель, снижает переобучение, улучшает качество на тестовых данных (ценой некоторого ухудшения на обучающих). Не выполняет отбор признаков (все коэффициенты остаются ненулевыми).

**7. Лассо Регрессия (Lasso Regression) - L1 Регуляризация**

**Принцип**: Тоже линейная модель (y^ = wx + b), но использует другой тип штрафа — пропорциональный сумме абсолютных значений (модулей) коэффициентов w (L1-норма).  
 Минимизировать (MSE + alpha \* Σ|w[i|)

Ключевое отличие от Ridge: L1-регуляризация может приводить к тому, что некоторые коэффициенты w становятся в точности равными нулю.

Выполняет автоматический отбор признаков: признаки, соответствующие нулевым коэффициентам, фактически исключаются из модели.

Может привести к более простым и интерпретируемым моделям, если действительно важна лишь часть признаков.

Параметр alpha: чем больше alpha, тем сильнее регуляризация, тем больше коэффициентов обнуляется.

Практика: Может требовать увеличения max\_iter (максимального числа итераций) для сходимости алгоритма обучения.

**8. Сравнение Ridge и Lasso**

**Ridge**: Сжимает коэффициенты к нулю, но не обнуляет их полностью. Хорош как общая техника регуляризации. Часто является методом выбора по умолчанию.

**Lasso**: Может обнулять коэффициенты, выполняя отбор признаков. Полезен, если есть подозрение, что многие признаки не важны, или если нужна очень простая, интерпретируемая модель.

**ElasticNet**: Комбинация L1 и L2 регуляризации, пытается взять лучшее от обоих миров (не рассматривается подробно в этой лабе, но упоминается).

**9. Кривые Обучения (Learning Curves)**

Назначение: Инструмент для анализа того, как качество модели (на обучающей и тестовой выборках) зависит от количества данных, используемых для обучения.

Польза: Помогают диагностировать переобучение (большой разрыв между кривыми) или недообучение (обе кривые показывают низкое качество), а также понять, поможет ли добавление новых данных улучшить модель.

**Пук**

**Линейные Модели для Бинарной Классификации:**

**Цель**: Разделить объекты на два класса (например, +1 и -1, или 1 и 0).

**Формула**: Сначала вычисляется взвешенная сумма признаков, точно так же, как в линейной регрессии:  
 Результат = w[0]\*x[0] + w[1]\*x[1] + ... + w[p]\*x[p] + b

**Принятие решения:** Вместо того чтобы использовать Результат напрямую как прогноз, его сравнивают с порогом, который для простоты обычно равен нулю:

Если Результат > 0, прогнозируем класс +1.

Если Результат < 0, прогнозируем класс -1.

**Ключевые отличия между алгоритмами:**

**Метрика качества** (Функция потерь / Loss function): Как алгоритм измеряет, насколько хорошо текущие w и b подходят к обучающим данным. Минимизировать просто количество ошибок напрямую сложно по математическим причинам, поэтому используются другие функции потерь (например, логистическая функция потерь, шарнирная функция потерь).

**Использование и тип регуляризации:**Применяется ли штраф к модели за сложность (большие коэффициенты w), и какой тип штрафа (L1 или L2).

**Основные Алгоритмы в Лабораторной:**

**Логистическая Регрессия** (LogisticRegression): Несмотря на слово "регрессия" в названии, это алгоритм классификации. Он использует логистическую функцию потерь.

**Линейный Метод Опорных Векторов** (LinearSVC - Linear Support Vector Classifier): Линейная версия мощного метода SVM. Использует шарнирную функцию потерь (hinge loss).

**Регуляризация в Классификации:**

L2 Регуляризация (по умолчанию): И LogisticRegression, и LinearSVC по умолчанию используют L2-регуляризацию, аналогичную Ridge регрессии. Она штрафует модель за большие значения коэффициентов w, заставляя их быть меньше по модулю, что помогает бороться с переобучением.

Параметр C: В этих моделях степень регуляризации контролируется параметром C. Важно: C обратно пропорционален alpha из Ridge/Lasso.

Большое значение C: Слабая регуляризация. Модель пытается максимально точно подогнаться под обучающие данные, уделяя внимание каждой точке. Может привести к переобучению.

Маленькое значение C: Сильная регуляризация. Модель ищет более простые решения с меньшими коэффициентами w, больше ориентируясь на "большинство" точек, а не на выбросы. Может привести к недообучению.

L1 Регуляризация: В LogisticRegression можно включить L1-регуляризацию (указав penalty="l1"). Как и в Lasso регрессии, она может обнулять некоторые коэффициенты w, тем самым выполняя отбор признаков. Это полезно для интерпретации модели, чтобы понять, какие признаки наиболее важны.

**Интерпретация Результатов и Графиков:**

Границы решений (**Рис. 1, 2**): Показывают, как модели разделяют классы в пространстве признаков. Прямая линия — характерная черта линейных классификаторов. Видно, как изменение C влияет на наклон и положение этой линии (при малом C линия более "общая", при большом C старается подстроиться под отдельные точки).

Качество (Правильность / Accuracy): Сравнивая правильность на обучающем и тестовом наборах при разных C, можно диагностировать недообучение (низкая правильность везде, часто при малом C), переобучение (высокая правильность на обучении, низкая на тесте, часто при большом C) или найти хороший компромисс.

Коэффициенты (**Рис. 3, 4**): Графики показывают величину коэффициентов w для каждого признака.

С L2 (**Рис. 3**): При увеличении C (уменьшении регуляризации) коэффициенты растут по модулю.

С L1 (**Рис. 4**): При уменьшении C (усилении регуляризации) всё больше коэффициентов становятся точно нулевыми. Это позволяет отобрать наиболее влиятельные признаки.

**Основной Вывод:** Линейные модели для классификации (LogisticRegression, LinearSVC) работают схожим образом с линейной регрессией, используя взвешенную сумму признаков. Ключевое отличие — использование порога для принятия решения о классе. Регуляризация (L1 и L2), контролируемая параметром C (обратным alpha), играет важную роль в настройке сложности модели, предотвращении переобучения и, в случае L1, в отборе признаков.

Пук

Эта лабораторная работа посвящена адаптации линейных моделей для задач мультиклассовой классификации, то есть когда классов больше двух.

**Ограничение Бинарных Моделей:**

Многие простые линейные классификаторы (как LinearSVC, рассмотренный ранее) изначально разработаны только для бинарной классификации (два класса). Логистическая регрессия является исключением и может напрямую обрабатывать несколько классов, но часто используется тот же механизм, что и для других моделей.

**Подход "Один против Остальных" (One-vs.-Rest, OvR):**

Это общий механизм, позволяющий использовать бинарный классификатор для решения мультиклассовой задачи.

Принцип: Если у нас K классов, мы строим K отдельных бинарных классификаторов.

Классификатор 1: Учится отличать Класс 1 от всех остальных классов (Класс 2, Класс 3, ...).

Классификатор 2: Учится отличать Класс 2 от всех остальных (Класс 1, Класс 3, ...).

... и так далее для каждого класса.

Результат обучения: Для каждого из K бинарных классификаторов мы получаем свой набор коэффициентов w и свою константу b.

**Получение Прогноза в OvR:**

Когда нужно классифицировать новую точку данных:

1. Эта точка подается на вход каждому из K бинарных классификаторов.
2. Каждый классификатор вычисляет значение по формуле: Значение\_класса\_i = w\_i[0]\*x[0] + ... + w\_i[p]\*x[p] + b\_i  
   (где w\_i и b\_i — параметры i-го бинарного классификатора). Это значение показывает, насколько уверенно i-й классификатор относит точку к "своему" классу (в противоположность "всем остальным").
3. "Победителем" объявляется тот класс, чей бинарный классификатор выдал наибольшее значение. Метка этого класса и становится итоговым прогнозом для точки.

**Структура Параметров :**

Поскольку мы строим по одному бинарному классификатору на класс, итоговая модель хранит параметры для всех них.

coef\_ (коэффициенты): Будет иметь форму (количество\_классов, количество\_признаков). Каждая строка содержит вектор коэффициентов w для одного из классов.

intercept\_ (константы): Будет одномерным массивом формы (количество\_классов,), где каждый элемент — это константа b для соответствующего класса.

**Визуализация (на примере 3 классов и 2 признаков):**

**Рис. 2:**Показывает три разделяющие линии, соответствующие трем бинарным классификаторам ("Класс 0 vs остальные", "Класс 1 vs остальные", "Класс 2 vs остальные"). Каждая линия пытается отделить точки "своего" класса от всех прочих.

Треугольник в центре: Область, где все три бинарных классификатора относят точку к категории "остальные". Какой класс будет присвоен точке в этой области? Тот, чья линия находится ближе всего к этой точке (или, эквивалентно, тот, чей классификатор дает наибольшее значение функции решения).

**Рис. 3:**Показывает итоговые области классификации. Пространство разделено на три региона. Границы между регионами формируются из частей трех исходных бинарных линий. Каждая точка в пространстве окрашена в цвет того класса, который ей присваивается по правилу "выбери класс с максимальным значением функции решения".

**Преимущества, Недостатки и Параметры (Общее для Линейных Моделей):**

Основной параметр: Параметр регуляризации (alpha в регрессии, C в LinearSVC/LogisticRegression). Большие alpha / маленькие C -> более простые модели (сильная регуляризация). Маленькие alpha / большие C -> более сложные модели (слабая регуляризация). Настраивается по логарифмической шкале.

**Выбор L1 vs L2:**

L1 (Lasso, penalty='l1'): Используется, если предполагается, что важны лишь некоторые признаки, или если нужна высокая интерпретируемость (L1 обнуляет неважные признаки).

L2 (Ridge, penalty='l2' - по умолчанию): Используется в большинстве случаев как стандартная техника регуляризации.

**Преимущества:**

* Быстро обучаются и прогнозируют.
* Хорошо масштабируются на большие и разреженные данные (много нулей). Есть оптимизированные решатели (solver='sag', SGDClassifier, SGDRegressor).
* Относительно легко интерпретировать прогноз (хотя не всегда легко понять, почему коэффициенты именно такие, особенно при коррелирующих признаках).

**Недостатки:**

* Могут быть слишком простыми для данных с нелинейными зависимостями, особенно в низкоразмерных пространствах.
* Интерпретация коэффициентов может быть сложной при высокой корреляции между признаками.

**Когда использовать:**Хороши для очень больших наборов данных или когда количество признаков превышает количество наблюдений. Часто служат хорошей отправной точкой.

**Цепочка Методов (Method Chaining):** Концепция: Многие методы в scikit-learn (например, fit) возвращают сам объект модели (self). Это позволяет вызывать методы последовательно в одной строке.