МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

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

**ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ**

# ЛАБОРАТОРНАЯ РАБОТА №4-6

по дисциплине

‘Системы искусственного интеллекта‘

*Выполнил:*
Студент группы P33312
Абульфатов Руслан
Мехтиевич
*Преподаватель:*
Кугаевских Александр
Владимирович

Санкт-Петербург, 2023

# Модуль 2.

## Лабораторная работа 1. Метод линейной регрессии

### Введение

- Получите и визуализируйте статистику по датасету (включая количество, среднее значение, стандартное отклонение, минимум, максимум и различные квантили).
- Проведите предварительную обработку данных, включая обработку отсутствующих значений, кодирование категориальных признаков и нормировка.
- Разделите данные на обучающий и тестовый наборы данных.
- Реализуйте линейную регрессию с использованием метода наименьших квадратов без использования сторонних библиотек, кроме NumPy и Pandas (для использования коэффициентов использовать библиотеки тоже нельзя). Использовать минимизацию суммы квадратов разностей между фактическими и предсказанными значениями для нахождения оптимальных коэффициентов.
- Постройте **три модели** с различными наборами признаков.
- Для каждой модели проведите оценку производительности, используя метрику коэффициент детерминации, чтобы измерить, насколько хорошо модель соответствует данным.
- Сравните результаты трех моделей и сделайте выводы о том, какие признаки работают лучше всего для каждой модели.

### Описание метода

Метод линейной регрессии - это статистический метод, используемый для определения связи между зависимой и независимыми переменными. Принцип работы метода заключается в построении линии наилучшего соответствия данных.

### Псевдокод метода

1. Подготовить данные: разделить данные на тренировочный и тестовый наборы.
2. Выбрать модель: определить вид модели линейной регрессии.
3. Обучить модель: подобрать параметры модели с использованием тренировочных данных.
4. Оценить модель: оценить точность модели на тестовом наборе данных.
5. Применить модель: использовать обученную модель для прогноза значений зависимой переменной.

# Результаты выполнения

**Модель по всем признакам**

```
y_pred, r2,sum_of_squares = perform_linear_regression(None, X_train, X_test, y_train, y_test)
print('Коэффициент детерминации:', r2)
print('Предсказания:', y_pred)
print('Сумма квадратов',sum_of_squares)
```

✓ 0.0s

```
Коэффициент детерминации: -4.8761773675185065
Предсказания: [107.62147574 136.78838895  56.39399907 ...  87.27853121  22.27258974
  77.27169669]
Сумма квадратов 4274147.598552551
```

**Модель по Previous Scores**

```
y_pred, r2,sum_of_squares = perform_linear_regression('Previous Scores', X_train, X_test, y_train, y_test)
print('Коэффициент детерминации:', r2)
print('Предсказания:', y_pred)
print('Сумма квадратов',sum_of_squares)
```

✓ 0.0s

```
Коэффициент детерминации: 0.8362130395777159
Предсказания: [44.66867552 70.00349758 54.80260434 ... 81.15081928 27.44099652
 72.03028334]
Сумма квадратов 119133.5114274736
```

**Модель по Hours Studied**

```
y_pred, r2,sum_of_squares = perform_linear_regression('Hours Studied', X_train, X_test, y_train, y_test)
print('Коэффициент детерминации:', r2)
print('Предсказания:', y_pred)
print('Сумма квадратов',sum_of_squares)
```

✓ 0.0s

```
Коэффициент детерминации: 0.13418859160569407
Предсказания: [46.7575746  60.68480336 49.54302035 ... 66.25569486 60.68480336
 57.89935761]
Сумма квадратов 629764.1341535407
```

**Модель по Previous Scores и Hours Studied**

```
y_pred, r2,sum_of_squares = perform_linear_regression('Hours Studied,Previous Scores', X_train, X_test, y_train, y_test)
print('Коэффициент детерминации:', r2)
print('Предсказания:', y_pred)
print('Сумма квадратов',sum_of_squares)
```

✓ 0.0s

```
Коэффициент детерминации: 0.9852961175000855
Предсказания: [36.04043077 75.7701531  49.0755955  ... 92.67988608 33.02070287
 74.949117  ]
Сумма квадратов 10695.144163585432
```

**Модель по Previous Scores, Hours Studied и Motivation**

```
y_pred, r2,sum_of_squares = perform_linear_regression('Hours Studied,Previous Scores,Motivation', X_train, X_test, y_train, y_test)
print('Коэффициент детерминации:', r2)
print('Предсказания:', y_pred)
print('Сумма квадратов',sum_of_squares)
```

✓ 0.0s

```
Коэффициент детерминации: 0.9855491500583068
Предсказания: [36.33500877 76.06565498 48.78494173 ... 92.39069988 32.73500531
 74.65883811]
Сумма квадратов 10511.096196099748
```

Можно заметить, что коэффициент детерминации сильно повысился за счет Previous Scores и Hours Studied. Из этого можно сделать вывод, что успеваемость зависит от того, какое количество часов он посвящает учёбе. Таким образом, чем больше ученик посвящает времени учебе и меньше внеклассовых занятий, тем выше его успеваемость. Также синтетический признак, который я добавил, а именно, мотивация поднимает коэффициент детерминации, следовательно, мотивация прямопропорциональна успеваемости.

### Примеры использования метода

Метод линейной регрессии представляет собой эффективный инструмент, применимый в различных контекстах. Например, его использование может быть направлено на прогнозирование объемов продаж, опираясь на информацию о рекламных затратах и других факторах, таких как сезон, погодные условия и прочее. Также, данный метод применяется для анализа взаимосвязей между разными переменными, такими как цена товара, его характеристики и уровень спроса на рынке. Выбор линейной регрессии обусловлен ее простотой, понятностью и высокой предсказательной способностью, что делает его ценным инструментом в анализе данных.

# Лабораторная работа 2. Метод k-ближайших соседей (k-NN)

## Введение

- Проведите предварительную обработку данных, включая обработку отсутствующих значений, кодирование категориальных признаков и масштабирование.
- Реализуйте метод k-ближайших соседей без использования сторонних библиотек, кроме NumPy и Pandas.
- Постройте две модели k-NN с различными наборами признаков:
  - Модель 1: Признаки случайно отбираются .
  - Модель 2: Фиксированный набор признаков, который выбирается заранее.
- Для каждой модели проведите оценку на тестовом наборе данных при разных значениях k. Выберите несколько различных значений k, например, k=3, k=5, k=10, и т. д. Постройте матрицу ошибок.

## Описание метода

Метод k-ближайших соседей используется для классификации объектов на основе их близости к примерам обучающей выборки. Основной принцип работы метода заключается в нахождении k ближайших соседей объекта и отнесении его к классу, который наиболее часто встречается среди этих соседей.

## Псевдокод метода

1. Для каждого объекта из обучающей выборки:
2. Вычислить расстояние между объектом из обучающей выборки и новым объектом.
3. Отсортировать объекты из обучающей выборки по возрастанию расстояния.
4. Выбрать k ближайших соседей.
5. Результирующее значение целевой переменной для нового объекта будет равно значению целевой переменной, которое имеют наиболее часто встречающиеся среди его k ближайших соседей.

## Результаты выполнения

**Постройте две модели k-NN с различными наборами признаков:**

*- Модель 1: Признаки случайно отбираются .*

*- Модель 2: Фиксированный набор признаков, который выбирается заранее.*

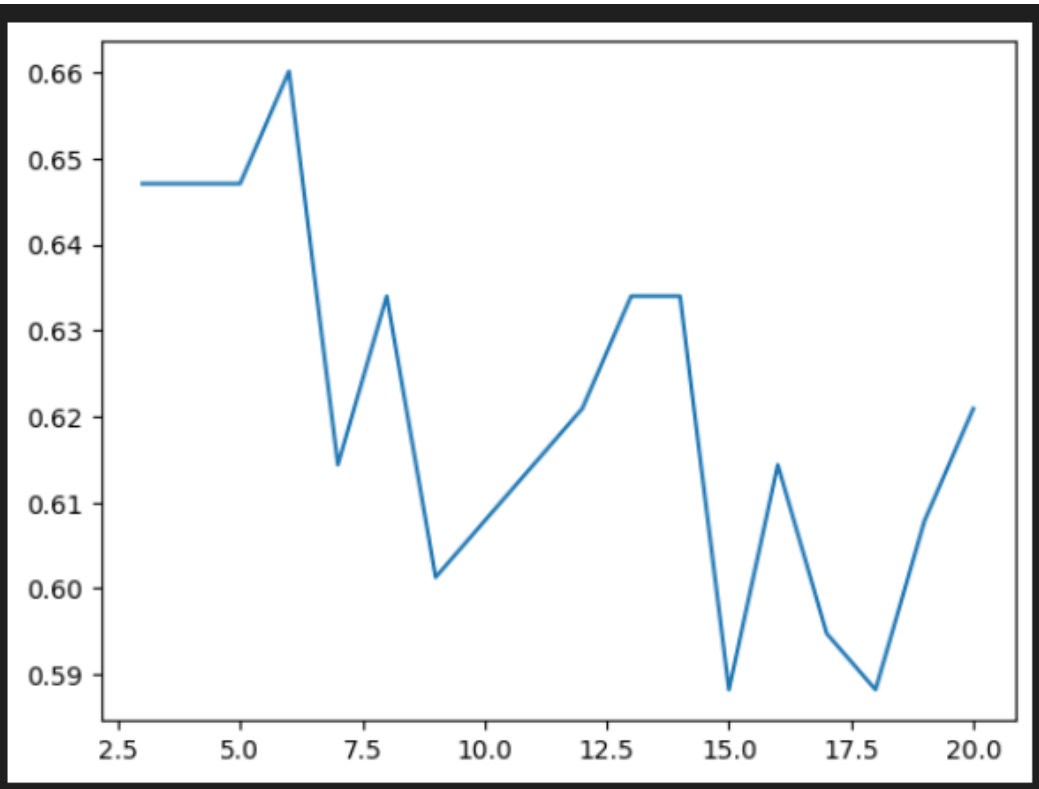
```
# Модель со случайными признаками
tags = ["Pregnancies", "Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI", "Pedigree", "Age"]
n = random.randint(1,8)
tags_1 = random.sample(tags, n)
print(tags_1)

X_test_rand = X_test[tags_1]
X_train_rand = X_train[tags_1]
X_test_rand=X_test_rand.to_numpy()
X_train_rand = X_train_rand.to_numpy()
k=[]
test_score = []
for i in range(3,21,1):
  clf = KNN(k=i)
  clf.fit(X_train_rand,y_train_np)
  y_pred = clf.predict(X_test_rand)
  show_cf_matrix(confusion_matrix(y_test_np, y_pred))
  print(f1_score(y_test_np,y_pred))
  test_score.append(f1_score(y_test_np,y_pred))
  k.append(i)

plt.plot(k,test_score)
plt.show
```

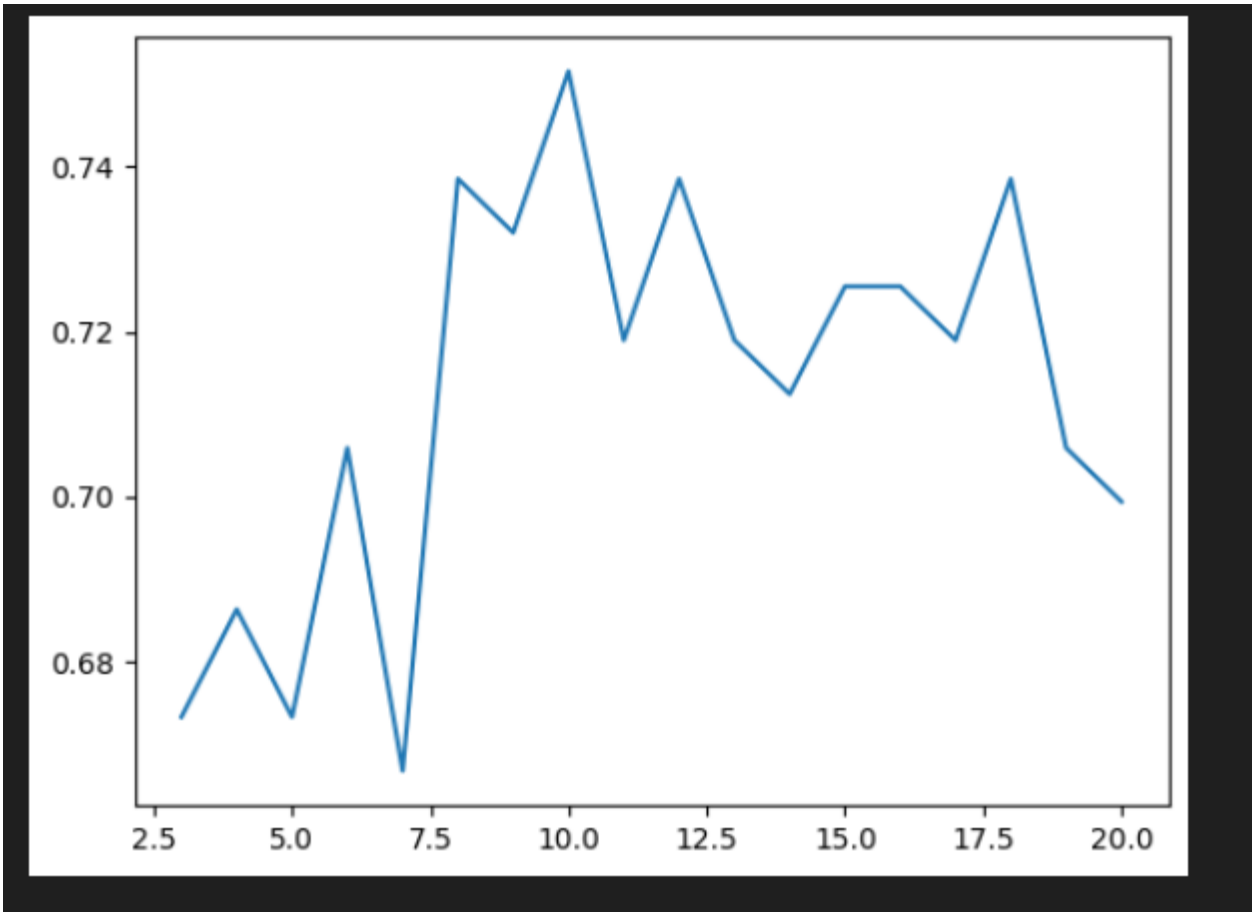
✓ 19.5s

```
['Insulin', 'Pedigree', 'Age', 'BloodPressure', 'Pregnancies', 'SkinThickness']
```



```python
# Модель с фиксированными признаками
k=[]
test_score = []
tags = ["Glucose","BloodPressure","BMI"]
X_test_fix = X_test[tags]
X_train_fix = X_train[tags]
X_test_fix=X_test_fix.to_numpy()
X_train_fix = X_train_fix.to_numpy()
k=[]
test_score = []
for i in range(3,21,1):
    clf = KNN(k=i)
    clf.fit(X_train_fix,y_train_np)
    y_pred = clf.predict(X_test_fix)
    show_cf_matrix(confusion_matrix(y_test_np, y_pred))
    print(f1_score(y_test_np, y_pred))
    test_score.append(f1_score(y_test_np,y_pred))
    k.append(i)

plt.plot(k,test_score)
plt.show
```



**Вывод:**

При увелечении количества ближайших соседей f1-score уменьшается. Оптимальным выбором считаю использовать 3-6 соседей.

### Примеры использования метода

Метод k-ближайших соседей может быть полезен в следующих ситуациях:

Когда у нас есть обучающая выборка, для которой известны значения целевой переменной, и мы хотим классифицировать новый объект.

Когда данные имеют сложную структуру и требуют нелинейной модели для классификации или регрессии.

# Лабораторная работа 3. Деревья решений

## Введение

1. Для студентов с четным порядковым номером в группе – датасет с классификацией грибов, а нечетным – датасет с данными про оценки студентов инженерного и педагогического факультетов (для данного датасета нужно ввести метрику: студент успешный/неуспешный на основании грейда)
2. Отобрать **случайным** образом sqrt(n) признаков
3. Реализовать без использования сторонних библиотек построение дерева решений (numpy и pandas использовать можно, использовать списки для реализации дерева - нельзя)
4. Провести оценку реализованного алгоритма с использованием Accuracy, precision и recall
5. Построить AUC-ROC и AUC-PR (в пунктах 4 и 5 использовать библиотеки нельзя)

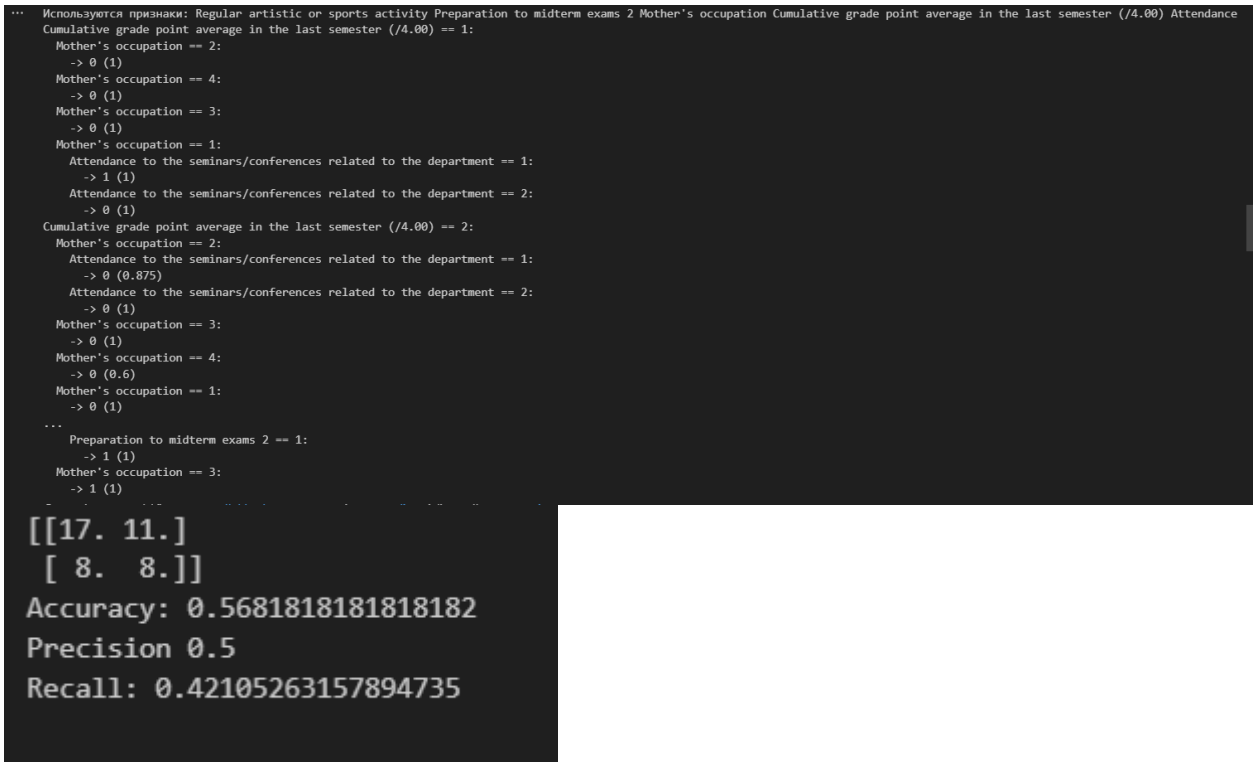
## Описание метода

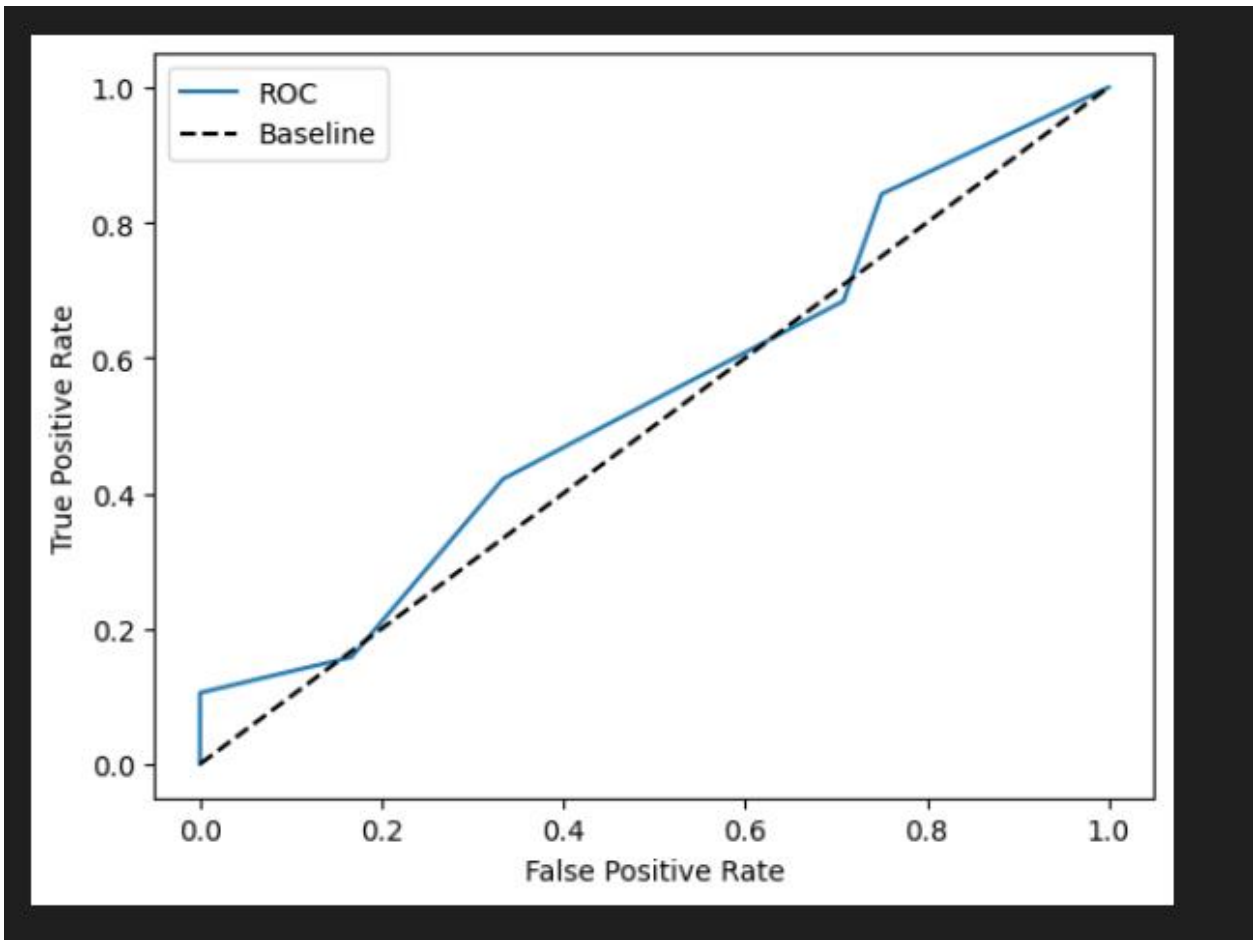
Метод деревьев решений является методом машинного обучения, который основывается на создании дерева, в котором каждый узел представляет условие или атрибут, а каждое ребро - результат этого условия. Дерево решений используется для прогнозирования или принятия решений на основе заданных данных. Он может применяться как для задач классификации, так и для задач регрессии.

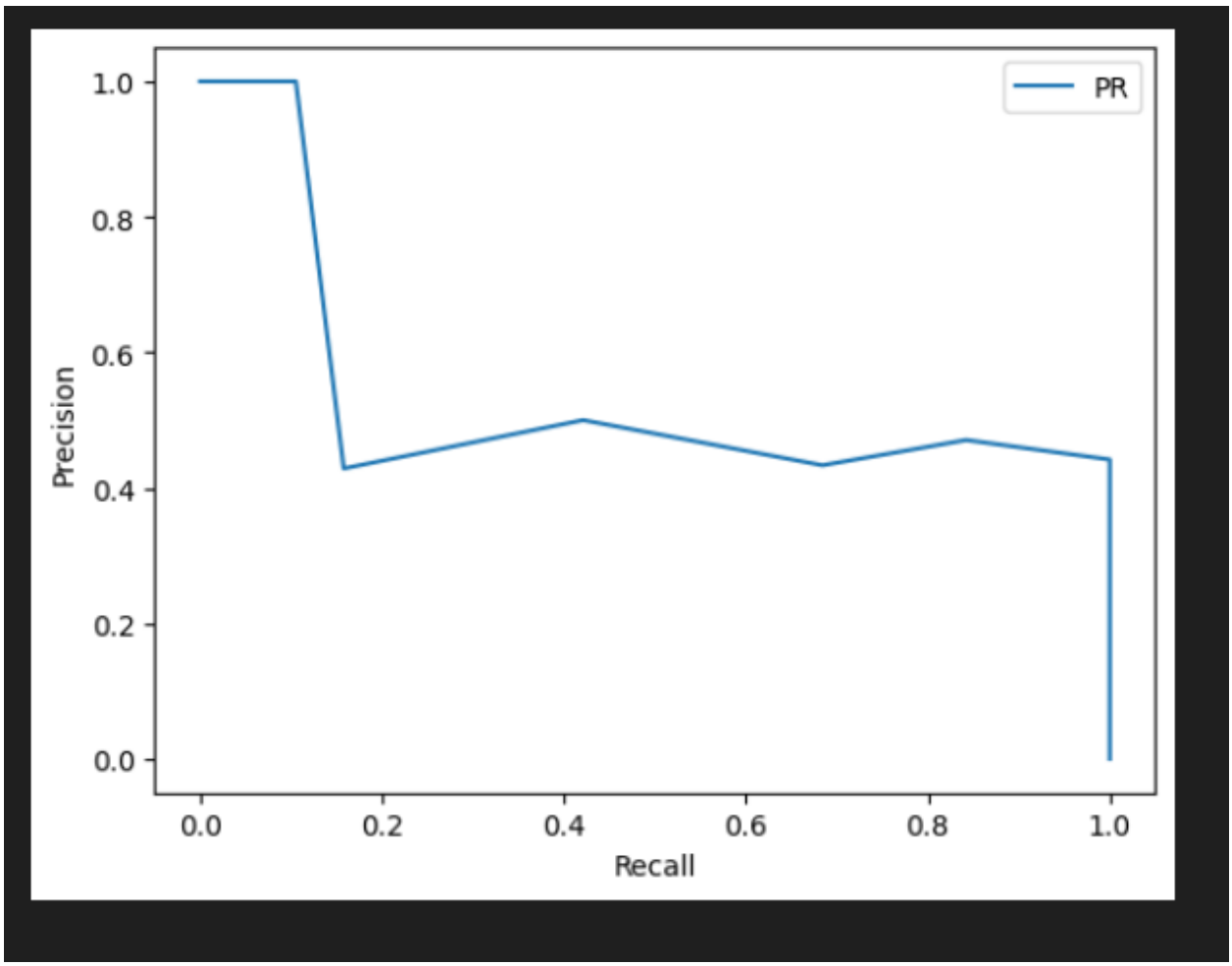
## Псевдокод метода

[A] Если все объекты в выборке относятся к одному классу, вернуть узел с этим классом
[B] Если все атрибуты уже рассмотрены, вернуть узел с наиболее часто встречающимся классом
[C] Иначе
  [D] Найти атрибут с наибольшим приростом информации
  [E] Создать узел для выбранного атрибута
  [F] Для каждого значения атрибута создать потомок на дереве
  [G] Рекурсивно применить алгоритм для новых потомков

# Результаты выполнения




ROC
Baseline
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0




## Примеры использования метода

Метод деревьев решений может быть полезен в следующих ситуациях:

Классификация пациентов на основе медицинских параметров для определения наличия или отсутствия конкретного заболевания.

Оценка вероятности успеха студентов на основе их академических показателей, учебных привычек и других факторов.

Классификация пользователей социальных медиа на основе их поведения, предпочтений и взаимодействия с контентом.

Классификация отзывов клиентов для выявления их настроений, удовлетворенности продуктом или услугой.

# Сравнение методов

## Сравнительный анализ методов

**Линейная регрессия:**

- Преимущества: Простота реализации и интерпретации, хорошая производительность на данных с линейной зависимостью, подходит для предсказания непрерывных значений.

- Ограничения: Линейная регрессия не справляется с нелинейными зависимостями, чувствительна к выбросам и шуму в данных.

- Ограничения: плохо работает с данными, имеющими сложную нелинейную структуру, требует линейной разделимости классов.

**Деревья решений:**

- Преимущества: может работать с любыми типами данных, обработка пропущенных значений и выбросов, хорошо интерпретируемый результат, легко обработать категориальные переменные.

- Ограничения: Склонны к переобучению на сложных данных, неустойчивость к небольшим изменениям в данных.

**Метод k ближайших соседей:**

- Преимущества: Простота реализации, хорошо работает на данных с нелинейной зависимостью, способен обрабатывать выбросы и шум в данных.

- Ограничения: требуется хранение всего обучающего набора данных, неэффективен при работе с большими объемами данных, требуется определение и настройка значения k.

## Примеры лучшего использования каждого метода

- Линейная регрессия может быть эффективна при предсказании цен на недвижимость, где зависимость между факторами и ценой может быть линейной.

- Деревья решений могут быть эффективны при принятии решений о предоставлении кредита, где нужно учитывать множество факторов.

- Метод k ближайших соседей может использоваться для классификации текстовых документов или обработки изображений с нелинейной структурой.

# Заключение

В зависимости от типа данных, сложности задачи и требований к интерпретируемости, каждый из этих методов имеет свои преимущества и ограничения. Важно выбирать метод, который наиболее подходит для конкретной задачи и обучать модель с использованием оптимальных гиперпараметров.

# Приложения

Код реализованных методов: https://github.com/ThisAster/Artificial-Intelligence-Systems