# Отчет по лабораторной работе №3
# по курсу «Машинное обучение»

Студент группы 8О-308 Баженова Надежда, № по списку 1.

Контакты: nad110508@gmail.com
Работа выполнена: 30.03.19

**1. Постановка задачи**

Требуется реализовать класс на выбранном языке программирования, который реализует один из алгоритмов машинного обучения. Обязательным является наличия в классе двух методов fit, predict. Необходимо проверить работу вашего алгоритма на ваших данных (на таблице и на текстовых данных), произведя необходимую подготовку данных. Также необходимо реализовать алгоритм полиномиальной регрессии, для предсказания значений в таблице. Сравнить результаты с стандартной реализацией sklearn, определить в чем сходство и различие ваших алгоритмов. Замерить время работы алгоритмов. Каркас класса на языке Python:

```
class MLAlgo(object):
def __init__(self,*pargs,**kwargs):
        pass
def fit(self,x,y):
        pass
def predict(self,x,y):
```

Алгоритмы на выбор:

1) Логистическая регрессия

2) KNN (K ближайших соседей)

3) SVM (Метод опорных векторов)

4) Рандомизированный лес

5) Наивный байесовский классификатор

6) Метод K-средних

Выбор алгоритма осуществляется по: номер_в_списке % 6 + 1

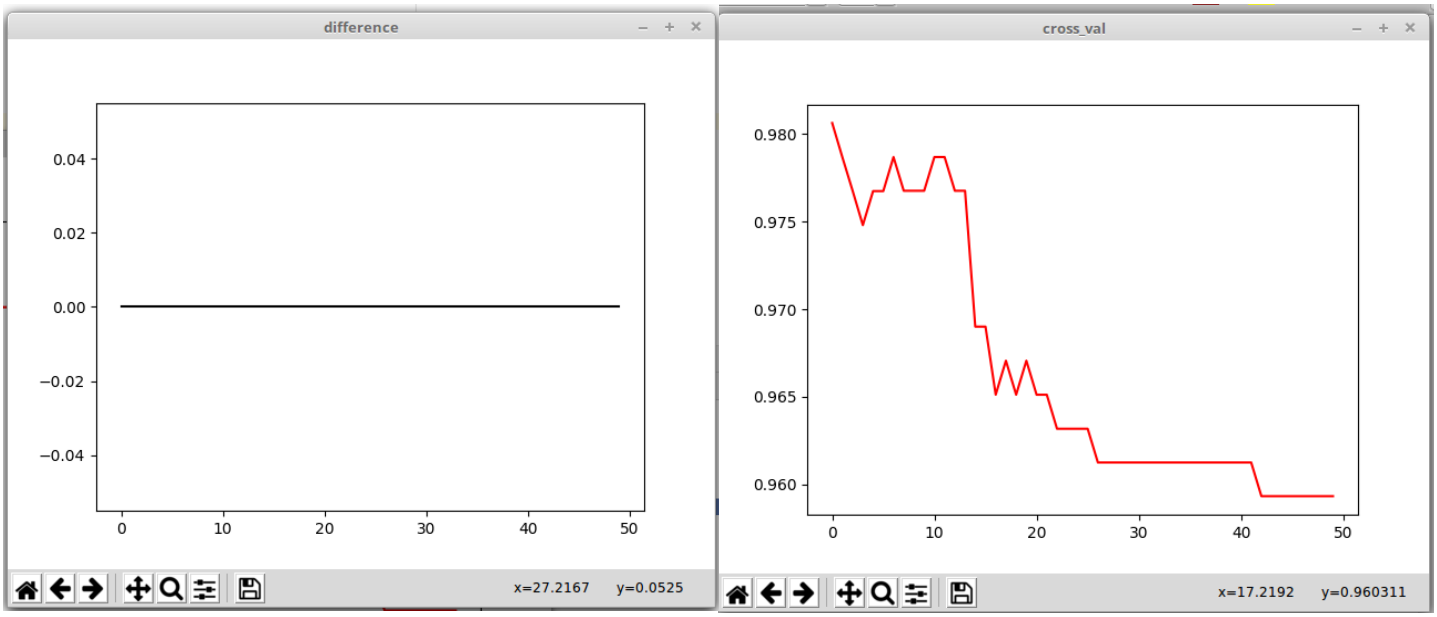
**Мой алгоритм: KNN**

3. Описание выполненной работы.

Я реализовала class knn, имеющий требуемые методы. При проверке результатов своей программы  я воспользовалась своими датасетами.

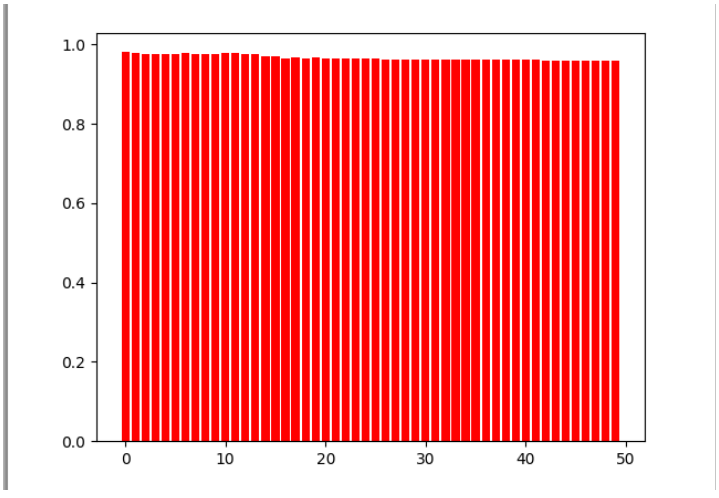
Для табличного датасета я применила knn-алгоритм в задачи классификации пожаров, произошедших в первой и второй половине года.

Я прогнала свой алгоритм и алгоритм, реализованный в sklearn.neighbors с разным количеством ближайших соседей и получила полное совпадение точности, что подтверждает правильность работы моего класса.




Так же с помощью кросс-валидации я нашла оптимальное число соседей для достижения хорошей точности.

На графике видно, что при увеличении количества соседей точность уменьшается. Однако диапазон значений не велик, и алгоритм показывает очень хорошие результаты(график ниже). Можно сделать вывод, что рассматриваемые классы расположены далеко друг от друга. И почти на пересекаются.



Для проверки алгоритма к ближайших соседей на текстовом датасете, я взяла новые данные, тк. Выбранный ранее датасет был не размечен по категориям и не давал возможности поставить задачу классификации.

Новый датасет:[https://ndownloader.figshare.com/files/5975967](https://ndownloader.figshare.com/files/5975967)

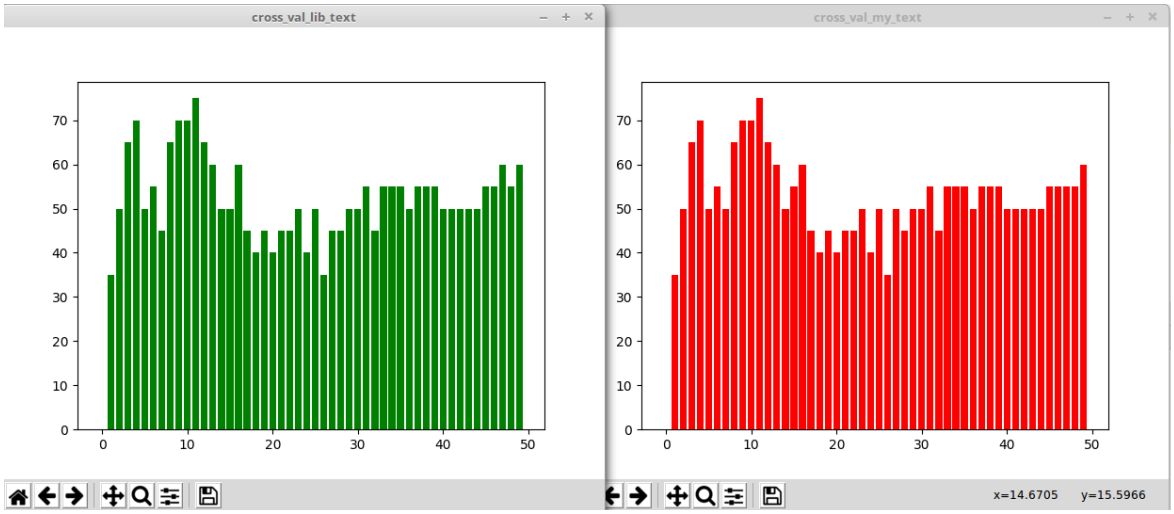
Я выбрала документы соответствующие 2м категориям:

```
'sci.med', 'rec.sport.baseball'
```

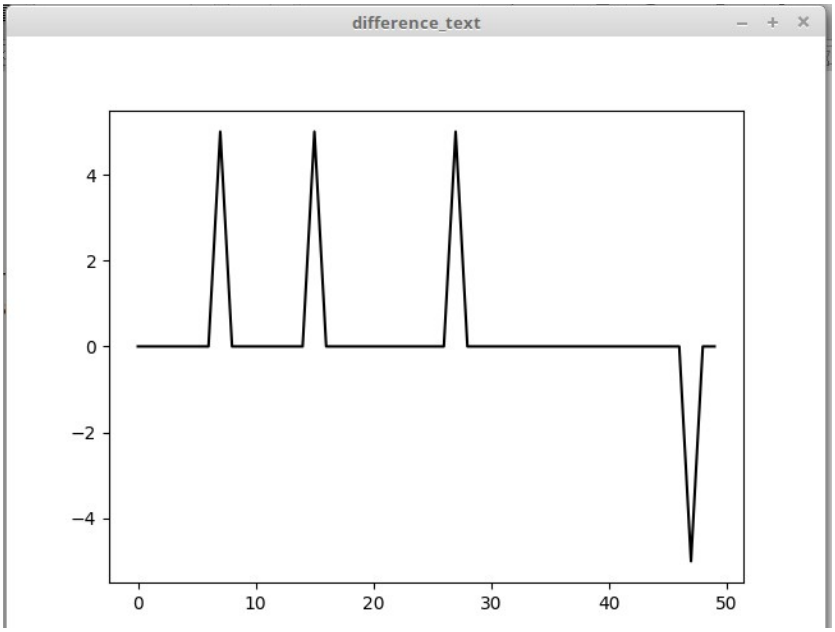
И поставила задачу классификации документов на 2 категории.

Обучив модель, я сформировала для каждого документа вектор признаков, состоящем из информации о частоте встречания слов из словаря в документе.

Далее я проделала тот же алгоритм тестирование knn, что и с таблицей.




Также я решила изобразить график отличия результатов точности 2х алгоритмов в зависимости от количества соседей(accuracy_my_knn - accuracy_lib_knn). Мы не видим идеального совпадения в некоторых ситуациях, однако отклонения не критичны.



Подводя итог, можно отметить, что алгоритм knn показывает меньшую точность для текстовых датасетов. Сложившая ситуация связана с распределением объектов(документов) в плоскости наших признаков. Скорее всего классы расположены близко к друг другу, что усложняет процесс угадывания результата

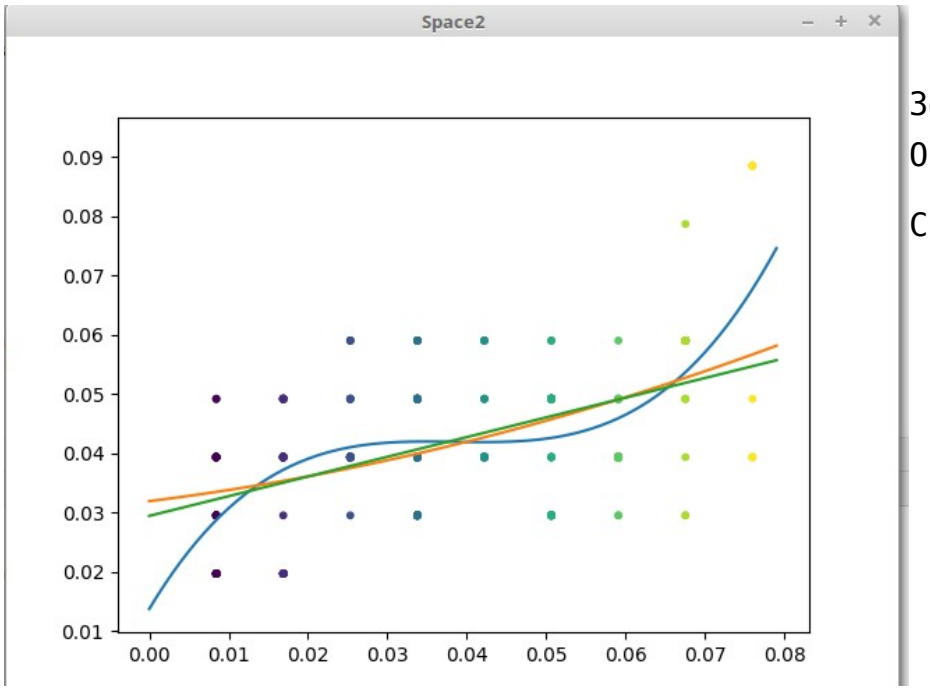
**Полиномиальная регрессия**

Для своего датасета я поставила задачу регрессии: узнать квадрат возгорания(y-неизвестная координата), при известной широте(x).

Таким образом, моя задача – найти функцию одного параметра f(x).

Посмотрим на распределение данных в плоскости x, y.

По графику видно, что распределение данных не линейное, поэтому линейная регрессия нам не подходит. Сведем нашу задачу к задачи построения функции многих переменных f(1, x, x^2,..). Посмотрим промежуточные результаты.




Зеленый – f(x)

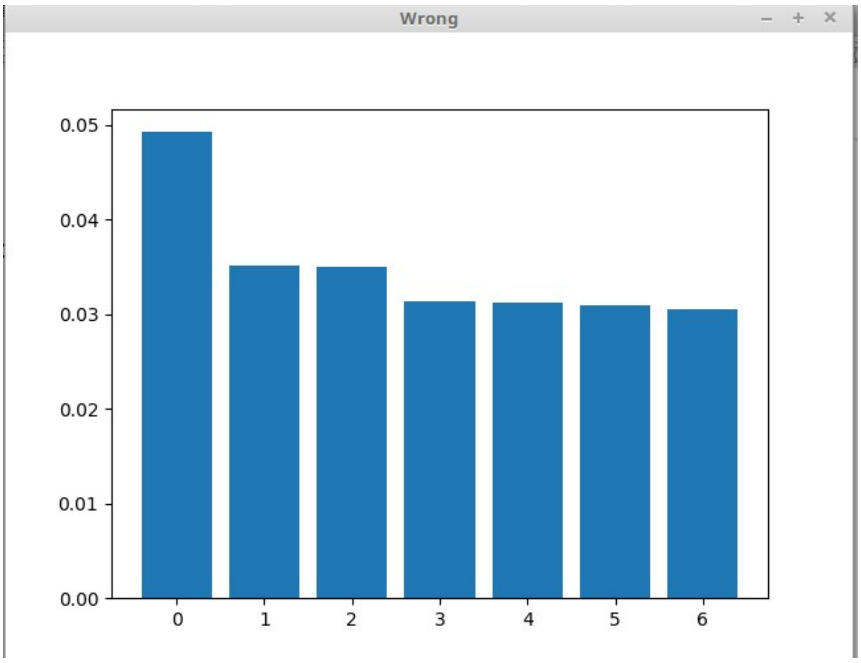
Оранжевый – f(x,x^2)

Синий – f(x,x^2,x^3)

Хорошо видно, что увеличение степени многочлена положительно сказывается на результате.

| Коэффициенты F(x) = b0+b1*x+b2*x^2….. | Ошибка, среднее отклонение(train/test) |
|---|---|
| [0.02941699 0.33246906] | 0.00010 / 0.00030 |
| [0.03190534 0.16728218 2.08652255] | 0.00010 / 0.00012 |
| [ 1.37077223e-02 2.23472335e+00 -5.84760434e+01 5.05627854e+02] | 0.00009 / 0.006789 |

Зависимость ошибки от степени полинома




Приемлемая степень полинома 3