# 

# 

# 

# Отчет по Лабораторной работе №1 и №2

“Деревья решений, Случайный Лес, Градиентный Бустинг”

## 

## 

## 

## 

## 

## 

## Курс: Интеллектуальные системы.

## Выполнил: Демянчук А.И. (АПИМ-17)

### 1. Описание структуры исходных данных и задачи в терминах предметной области и машинного обучения

Задача относится к области сельского хозяйства. Данные представляют собой набор характеристик трех различных сортов пшеницы (Kama, Rosa и Canadian) по 70 элементов каждого вида в выборке. Визуализация внутренней структуры зерна была проведена при помощи техники мягкого рентгеновского излучения. Изображения структуры зерен были сохранены на рентгеновские пластины Kodak. Зерна пшеницы были получены из экспериментальных полей Института Агрофизики Польской Академии Наук (г. Люблин).

#### Формальное описание задачи в терминах машинного обучения

Задачей является: разработать алгоритм классификации зерен пшеницы трех сортов (Kama, Rosa и Canadian) на основе семи параметров измерений зерна.

Класс задачи: задача классификации.

Исходные признаки:

1. Поверхность (A)

2. Периметр (P)

3. Компактность (C = 4\*pi\*A/P^2)

4. Длина зерна

5. Ширина зерна

6. Коэффициент асимметрии

7. Длина бороздки зерна

Целевая переменная:

Сорт зерна

1. Cama

2. Rosa

3. Canadian

### 2. Результаты предварительного анализа и визуализации исходных признаков и целевой переменной

Все признаки представлены вещественными переменными.

Классы: три дискретных класса. Тип данных Object в признаках Compactness и Kernel Width наводит на мысль о пропущенных данных. Некоторые данные в признаках Compactness и Kernel Width заменены на знаки вопроса. Эти данные найдены и заменены.

Проведен корреляционный анализ признаков. Площадь, периметр, длина зерна и длина бороздки имеют прямую зависимость. Кроме того при построении boxplot отмечено, что площадь, периметр и длина зерна лучше всего делят данные на классы.

### 3. Подготовка данных. Обучение и тестирование моделей

Проведена подготовка данных, обучение и тестирование моделей.

Лучший критерий разбиения - энтропия.

Количество признаков (max features) - лучший результат при использовании всех семи признаков.

Оптимальная глубина дерева - 7.

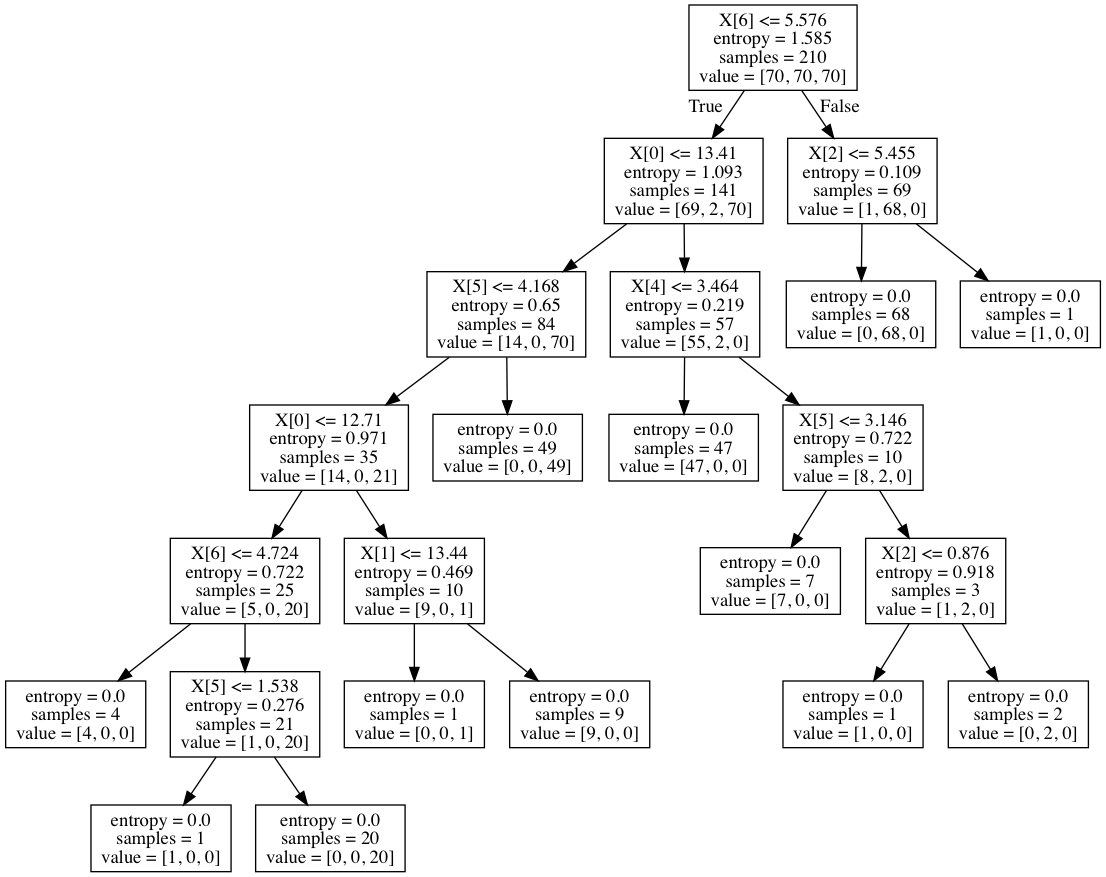
Код с оптимальными параметрами:

clf = DecisionTreeClassifier(criterion='entropy', max\_depth=7, max\_features=7, random\_state=random\_state)

scores = cross\_val\_score(clf, X, y, cv=5)

scores.mean()

**Визуализация работы дерева с оптимальными параметрами.**



К отчету приложен файл Lab1.ipynb с детальной визуализацией вышесказанного.

Проведено сравнение метода дерева решений с методом логистической регрессии.

### 4. Выводы

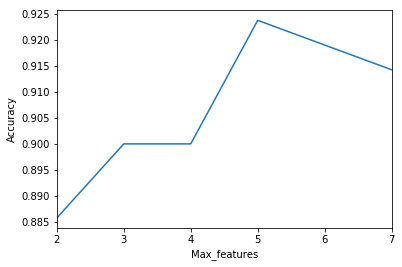
Проведен анализ задачи классификации и решение ее с использованием метода машинного обучения деревья решений. Входные данные (признаки) полностью числовые с минимальными пропусками в данных (не более 2%). При первичном анализе выделены признаки, которые оптимально делили данные на классы (площадь, периметр и длина зерна). Проведено пробное построение деревьев решений с использованием различных параметров. Лучший результат показало дерево, в котором использованы следующие параметры: критерий - энтропия, максимальной число признаков - 7, максимальная глубина дерева - 7. При выбранных параметрах достигнута точность классификации - 0.914. Кроме деревьев решений для сравнения использован метод решения задач классификации: логистическая регрессия. Этот метод был выбран с учетом картины входящих данных (небольшое количество признаков и тип признаков - вещественные). Со стандартными методами оптимизации и регуляризации достигнута высокая точность классификации - 0.957. Значение model.score больше чем cros\_val\_score.mean, потому что несмотря на использование кросс-валидации, обучение модели (fit) в итоге дает нам только лучшие из полученных весов. Метод model.score считает среднее значение model.predict(X) по отношению к y и значит находит score при использовании лучшего веса. cros\_val\_score же считает среднее всех score полученных при кросс-валидации.

## Часть 2.

## Случайный лес.

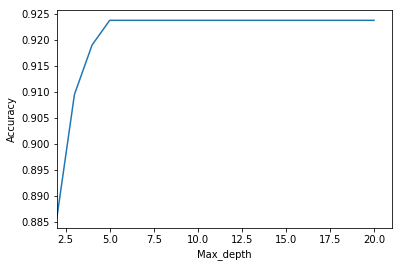
Проведен последовательный анализ параметров для классификатора Случайный лес.

Подобрано оптимальное число признаков = 5.



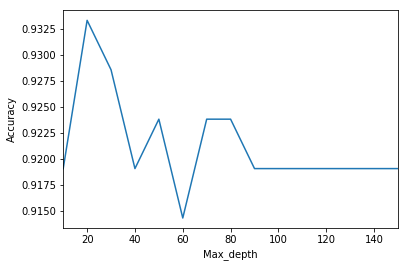
Далее проведен перебор параметров min\_samples\_leaf и min\_sample\_split. По полученным данным оптимальное значение Min\_samples\_leaf: 1 или 3 при средних значениях min\_samples\_split (приблизительно от 10 до 20). При этом точность на валидации несколько снижается (до 0.92).

Следующим шагом получено подходящее значение max\_depth (максимальной глубины дерева).



Оптимальная глубина - 5.

Следующим шагом получено оптимальное количество деревьев для Случайного Леса.



Лучший выбор - 20 деревьев на классификатор.

Код с результатом применения оптимальных параметров:

rfc = RandomForestClassifier(n\_estimators=20,max\_features=5,random\_state=random\_state)

score = cross\_val\_score(rfc, X, y, cv=5, n\_jobs=-1)

score.mean()

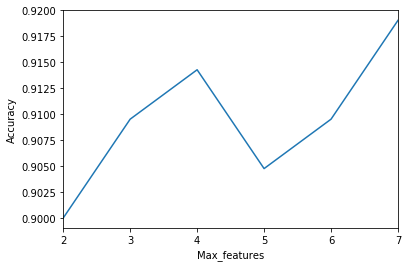
Итоговый набор параметров и точность для классификатора:

(RandomForestClassifier(bootstrap=True, class\_weight=None, criterion='gini',  
 max\_depth=5, max\_features=5, max\_leaf\_nodes=None,  
 min\_impurity\_split=1e-07, min\_samples\_leaf=1,  
 min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,  
 n\_estimators=20, n\_jobs=-1, oob\_score=False, random\_state=11,  
 verbose=0, warm\_start=False), 0.93333333333333335)

## Градиентный бустинг.

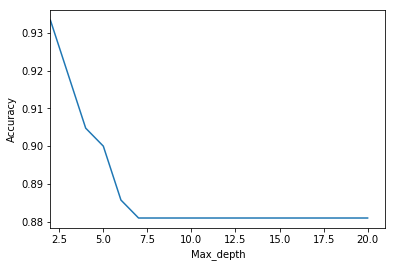
Проведен анализ параметров для градиентного бустинга.

Параметры max\_features: оптимальное количество признаков - 7

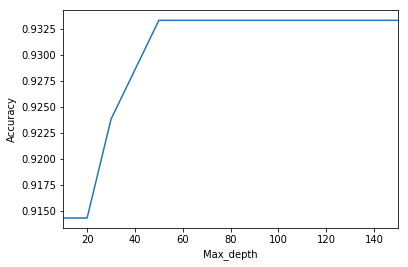


Оптимальное значение для аргумента min\_samples\_leaf (минимальное количество сэмплов необходимое для создания листа-узла) в сочетании с различными значениями min\_samples\_split (минимальное число сэмплов необходимое для расщепления узла). Наилучший результат получен при минимальных значениях min\_samples\_leaf.

Подходящее значение max\_depth (максимальной глубины дерева). Оно равно 2.



Оптимальное количество деревьев для Градиентного Бустинга = 50.



Код использования классификатора с оптимальными параметрами:

gbc = GradientBoostingClassifier(n\_estimators=50,

max\_depth=2,

min\_samples\_leaf=1,

max\_features=7,

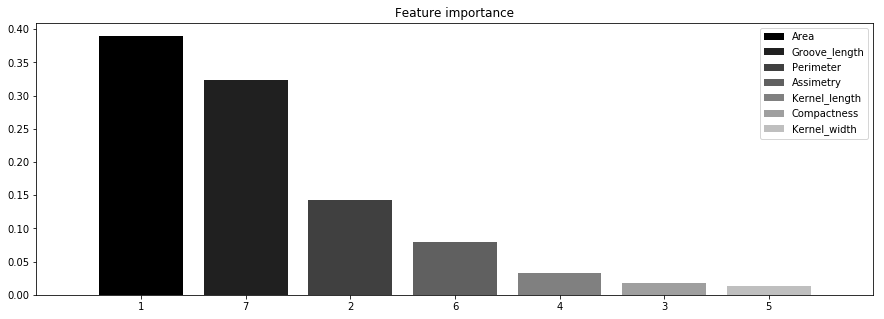
random\_state=random\_state)

score = cross\_val\_score(gbc, X, y, cv=5, n\_jobs=-1)

score.mean()

Результат: 0.9333.

Далее проанализирована важность признаков при использовании их в вышеуказанных классификаторах. Получены следующие результаты.



Результаты: Отдельное дерево = 0.914, Градиентный Бустинг = 0.933, Случайный лес = 0.933, Логистическая регрессия = 0.957.

## Выводы.

Учитывая высокую склонность деревьев решений к переобучению и соответственно худшей точности на тестовых или валидационных данных (плохо генерализуют) были предложены методы которые улучшают возможности деревьев решений. Мы рассмотрели RandomForest - который по существу является вариантом Bagging (Bootstrap Agregation). Данный метод снижает Variance и тем самым уменьшает переобучение, использую большое количество деревьев обучающихся на разных подвыборках основной выборки (созданных методом bootstrap).

В RandomForest классификаторе для того чтобы еще уменьшить корреляцию между деревьями для каждого расщепления используется случайный набор признаков m из общего числа признаков n (m < n).

Идея градиентного бустинга заключается в том, чтобы также использовать большое количество деревьев. Но в качестве обучающих данных для следующего дерева подается данные с учетом ошибки, допущенной на предыдущем дереве. Таким образом каждому дереву достаются все данные и большее внимание они обращают на данные с большей ошибкой.

Исследуя наши данные мы получили результаты, которые в большей мере подтверждают теорию. Хотя Случайный Лес и оказался равноценным Градиентному Бустингу.

При этом самым подходящим методом оказалась все же логистическая регрессия.