Школа молекулярной и теоретической биологии
V сезон

Опыты Быстротекущей Жизни

М. А. Ройтберг, Евгений Баулин, Алексей Драль

при участии Василия Рогова

Слушатели: ***Даниил Бобровский, Даша Быкова, Маша Селифанова, Артемий Юшкевич, Денис Яроцкий***

Барселона 2-19 августа 2016 г.

ЧАСТЬ 1.

Знакомство с теорией вероятности.

Занятие 1. 7.08.16

1. Вероятностное пространство

1.1. Исходы.

Рассмотрим конечное множество; его элементы называются *элементарными исходами* или просто *исходами.*

Рассматривая конкретную задачу, мы фиксируем определенное ПЭС и работаем только с ним.

Пример 1. Монетка. Состояния: {Орел, Решка}. Вероятности:

Состояния	***Орел***	***Решка***
Вероятности	0.5	0.5

Пример 2. Кубик.

Состояния	***1***	***2***	***3***	***4***	***5***	***6***
Вероятности	1/6	1/6	1/6	1/6	1/6	1/6

Пример 3. Две монетки.

2-я

1-я		***Орел***	***Решка***
	Орел	ОО	ОР
	Решка	РО	РР

Состояния

2-я

1-я		***Орел***	***Решка***
	Орел	1/4	1/4
	Решка	1/4	1/4

Вероятности

Вероятность того, что на 1-й монетке орел: ¼ +¼ = ½

Пример 4. Гнутая монетка.

Состояния	***Орел***	***Решка***
Вероятности	0.9	0.1

Событие – подмножество множества исходов. Над событиями можно выполнять теоретико-множественные операции – пересечение («*одновременное наступление событий*»), объединение («*наступление хотя бы одного из событий*»), дополнение («*наступление противоположного события*»).

1.2. Вероятности.

Каждому исходу приписана *вероятность* – число между 0 и 1.

Сумма вероятностей всех исходов равна 1.

Вероятность события - сумма вероятностей входящих в него исходов. Вероятность события А обычно обозначается *P(A)* или *Prob(A).*

Вероятностное пространство = множество исходов + вероятности исходов

Пример 5. Две странные монетки.

1-я \ 2-я	*Орел*	*Решка*
Орел	ОО	ОР
Решка	РО	РР

а) Состояния

1-я \ 2-я	*Орел*	*Решка*
Орел	0.3	0.1
Решка	0.5	0.1

б) Вероятности

Рис.1.

В вероятностном пространстве 4 элементарных исхода (см. Рис 1а); их вероятности даны на рис. 1б.

Рассмотрим 4 события в этом вероятностном пространстве (см. рис.2):

A. «На 1-й монетке выпал орел». Оно состоит из двух элементарных событий: ОО и ОР. Вероятность: *P(A)* = 0.3+0.1 = 0.4
B. «На 1-й монетке выпала пешка». Оно состоит из двух элементарных событий: РО и РР. Вероятность: *P(B)* = 0.5+0.1 = 0.6
C. «На 2-й монетке выпал орел». Оно состоит из двух элементарных событий: ОО и РО. Вероятность: *P(C)* = 0.3+0.5 = 0.8
D. «На 1-й монетке выпала пешка». Оно состоит из двух элементарных событий: ОН и РР. Вероятность: *P(D)* = 0.1+0.1 = 0.2

1-я \ **2-я**	*Орел*	*Решка*	**1-я м-ка**
Орел	0,30	0,10	**0,40**
Решка	0,50	0,10	**0,60**
2-я м-ка	**0,80**	**0,20**	

Рис.2.

Аналоги понятия вероятность – мера, площадь, объем, масса, цена.

1.3. Теория и жизнь.

Что мы, неформально говоря, понимаем под случайностью. Происходят события, про которые мы думаем, что они «одного типа» (на нашем языке – исходы из одного множества исходов). При этом:

1) Мы не можем предсказать, какой исход наступит.
2) Если проводить много одинаковых (независимых) испытаний, мы получим последовательность различных исходов. В частотах, с которыми появляются разные исходы, можно увидеть закономерности.

Есть две причины возникновения случайности: (1) квантовая физика; (2) слишком сложное явление, зависящее от большого количества факторов – выделяем основные, остальное рассматриваем как случайность («хаос»). Примеры: бросаем монетку, бросаем кубик, вызываем датчик случайных чисел.

Пример закономерности для частот: с увеличением количества испытаний частота каждого исхода стремится к определенному пределу. Для различных конкретных явлений этот факт можно проверить наблюдениями.

В нашей модели то, что предел существует и равен именно заданной в модели вероятности исхода – это теорема (закон больших чисел).

Вопрос: Допустим, мы провели наблюдения и предел не совпал с вероятностью. ~~Или его не существует~~. Что это означает?

Ответ: Значит, модель неадекватная.

2. Случайная величина

2.1. *Случайная величина* (с.в.) – это числовая функция, определенная на элементарных событиях. Т.е. каждому э.с. сопоставляется некоторое число.

Можно представить себе, что мы играем в азартную игру. Значение с.в. определяет наш выигрыш (при отрицательном значении – проигрыш) при наступлении конкретного события.

Примеры.

1. Монетка А

Состояния	***Орел***	***Решка***
Вероятности	1/2	1/2
Значение с.в.	0	1

Монетка Б

Состояния	***Орел***	***Решка***
Вероятности	1/2	1/2
Значение с.в.	-5	1

2. Кубик А

Состояния	***1***	***2***	***3***	***4***	***5***	***6***
Вероятности	1/6	1/6	1/6	1/6	1/6	1/6
Значение с.в.	***1***	***2***	***3***	***4***	***5***	***6***

Кубик Б

Состояния	***1***	***2***	***3***	***4***	***5***	***6***
Вероятности	1/6	1/6	1/6	1/6	1/6	1/6
Значение с.в.	***-1***	***2***	***0***	***0***	***5***	***-6***

2.2. Среднее значение (математическое ожидание)

Говоря неформально, среднее значение случайной величины – это среднее значение ее значений, полученное после большого количества реализаций соответствующего вероятностного пространства.

Поясним, почему мы думаем, что это значение будет примерно одинаковым при разных последовательностях реализаций.

Пусть $\{A_1, \ldots, A_K\}$ – количество элементарных событий в в.п.; N – длина последовательности реализаций (количество событий в ней). Тогда доля

Если все исходы равновероятны, то математическое ожидание с.в. – это среднее арифметическое ее возможных значений. Если не равновероятны, то нужно учитывать вероятности.

Пусть X – случайная величина; вероятностное пространство содержит N э.с.; p_k - вероятность k-го э.с.; X_k – значение с.в. на нем (k = 1,…, N). Тогда

$$M(X) = X_1*p_1 + X_2*p_2 + \ldots + X_N*p_N \qquad (*)$$

Пример НЕ НАПИСАН ☹

Если с.в. принимает одно и то же значение на нескольких э.с., то в формуле (*) удобно сделать – группировку – собрать вместе все слагаемые, соответствующие одному и тому же значению с.в.

Получим такую формулу.

Пусть случайная величина X принимает R различных значений: $Z_1, Z_2, \ldots, Z_R$

Пусть q_k – это вероятность того, что с.в. X принимает значение X_k, т.е. сумма вероятностей тех э.с., на которых с.в. X принимает значение Z_k (k = 1, …, R). Тогда:

$$M(X) = Z_1*q_1 + Z_2*p_2 + \ldots + Z_R*p_R \qquad (**)$$

Пример. НЕ НАПИСАН ☹

Т.к. в последовательности реализаций вероятностного пространства доля каждого исхода примерно равна его вероятности, то **среднее арифметическое соответствующих значений с.в. примерно равно математическому ожиданию с.в.**

Таблица, в которой перечислены все возможные значения с.в. и вероятности, с которыми принимаются эти значения, называется ***законом распределения с.в.***

Чтобы узнать м.о. с.в. достаточно знать ее закон распределения.
Про вероятностное пространство можно ничего не знать.

3. Независимые события

Продолжение следует

Занятие 2. 08.08.16

Неформально говоря, мы называем события A и B независимыми, если знание о том, что событие A произошло ничего не добавляет к нашим знаниям о том, произойдет ли событие B.

В нашей модели это выглядит так.

1) Ничего не знаем про A. Вероятность того, что случится B равна P(B).

2) Знаем, что A произошло. Вероятность того, что B тоже произойдет равна P(AB). Вероятность того, что B произойдет при условии, что A произошло (говорят: *условная вероятность*) равна P(AB)/P(A) [тут нужна картинка!]
3) Требование независимости:

$$P(AB)/P(A) = P(B)$$

Обычно записывают так:

$$\mathbf{P(AB) = P(A)*P(B)}$$

Примеры с двумя монетками – ну вы помните ☺

4. Преобразования случайных величин

Случайная величина – это функция, определенная на множестве исходов. Со случайными величинами можно обращаться как с обычными функциями – складывать, умножать и т.п. Вот примеры:

Состояния	***1***	***2***	***3***	***4***	***5***	***6***
Вероятности	1/6	1/6	1/6	1/6	1/6	1/6
Значение с.в. X	***1***	***2***	***3***	***4***	***5***	***6***
Значение с.в. Y	***6***	***5***	***4***	***3***	***2***	***1***
2*X	2	4	6	8	10	12
X+10	11	12	13	14	15	16
X^2	1	4	9	16	25	36
X+Y	7	7	7	7	7	7
X*Y	6	10	12	12	10	6

5. Свойства математического ожидания

Пусть X, Y – случайные величины, определенные на некотором вероятностном пространстве; a - число. Верны следующие утверждения.

1) M(aX) = a*M(X)
2) M(X+Y) = M(X)+M(Y)

Эти два свойства называются *линейностью.*

3) M(X+a) = a+M(X)
4) M(X-M(X)) = 0

Переход от с.в. X к с.в. X-M(X) называется *центрированием.*

5) Если для всех возможных исходов значение с.в. X не больше, чем значение с.в. Y, то

$$M(X) \le M(Y)$$

С.в. X и Y называются *независимыми*, если независима любая пара событий вида {X=a} и {Y=b}.

6) Пусть случайные величины X и Y независимы. Тогда

$$M(X*Y) = M(X)*M(Y)$$

Доказательство. *Не успел* ☹

6. Разброс значений случайной величины. Дисперсия

Продолжение следует

Занятия 3 и 4. 09.08 – 10.08.

На занятии 3 основное время заняла практическая работа в системе Ipython notebook. На занятии 4 материал про дисперсию был повторен.

1.1. Что такое дисперсия

Дисперсией случайной величины X называется число $M(X-M(X))^2$). т.е. математическое ожидание квадрата *центрированной величины* $X_C = X-M(X)$, рис.1.

Состояния	***A***	***B***	***C***	***D***	***E***	***F***	М.О.
Вероятности	1/6	1/6	1/6	1/6	1/6	1/6	
Значение с.в. X	***12***	***10***	***8***	***6***	***4***	***2***	***7***
X_C =X - M(X)	5	3	1	-1	-3	-5	0
(X-M(X))^2	25	9	1	1	9	25	70/6
D(X)							**70/6**

Рис.1.

Дисперсия служит мерой разброса значений случайной величины. Учитывается не только, насколько значения могут отличаться от среднего, но и с какой вероятностью происходят отклонения. Почему в качестве меры разброса берут м.о. от $(X-M(X))^2$, а не, скажем, от $|X-M(X)|$ станет понятно позже.

1.2. Свойства дисперсии

Ниже c, a, b, k – постоянные величины; X, Y – случайные величины; m_X, m_Y – их математические ожидания.

1) $D(c) = 0$
2) $D(X+c) = D(X)$
3) Доказательство. $(X+c) – M(X+c) = X+c – (M(X)+c))= X-M(X)$. И т.д.
4) $D(X) = M(X^2) – (m_X)^2$.
   Доказательство. $(X-M(X))^2 = (X- m_X)^2 = X^2 – 2*m_X*X + (m_X)^2$. Поэтому
   $M((X-M(X))^2) = M(X^2 – 2*m_X*X + (m_X)^2) = M(X^2) – M(2*m_X*X) + M((m_X)^2) =$
   $= M(X^2) – 2*m_X*M(X) + (m_X)^2 = M(X^2) – 2*m_X* m_X + (m_X)^2 =$
   $= M(X^2) – (m_X)^2$
5) $D(kX) = k^2*D(X)$

Доказательство. $M((kX)^2) – (M(kX))^2 = M(k^2*X^2) – (k*M(X))^2 =$
$= k^2*M(X^2) – k^2*(M(X))^2 = k^2*(M(X^2) – (M(X))^2) = k^2*(D(X))$

6) Пусть случайные величины X и Y независимы. Тогда
$D(X+Y) = D(X) + D(Y)$

Доказательство. Пусть $M(X) = m_X$; $M(Y) = m_Y$. Имеем:
$M((X+Y)^2) = M(X^2 + 2XY + Y^2) = M(X^2) + M(2XY) + M(Y^2) =$
$= M(X^2) +2* M(XY) + M(Y^2) =$
[т.к. X и Y независимы, то $M(X*Y) = M(X)*M(Y)$]
$= M(X^2) +2*M(X)*M(Y) + M(Y^2) = M(X^2) +2*m_X*m_Y + M(Y^2)$

Далее.
$(M(X+Y))^2 =(M(X) + M(Y))^2 = (m_X+m_Y)^2 = (m_X)^2 + 2*m_X*m_Y + (m_Y)^2$

$D(X+Y) = M((X+Y)^2) – (M(X+Y))^2 =$
$= (M(X^2) + 2*m_X*m_Y + M(Y^2)) – ((m_X)^2 + 2*m_X*m_Y + (m_Y)^2) =$
$= (M(X^2) - (m_X)^2) + (M(Y^2) - (m_Y)^2) = D(X) + D(Y)$, ч.т.д.

(здесь начинается новый материал занятия 4, 10.08).

7. Нормирование и центрирование.

Утверждение 1. (следствие свойства 5). Пусть $D(X) = (\sigma_X)^2$. Тогда дисперсия с.в. X/σ_X равна 1.

Доказательство – ясно.

Величина $\sigma_X = sqrt(D(X))$ называется *среднеквадратическим отклонением* с.в. X.

Деление с.в. X на ее среднеквадратическое отклонение называется *нормировкой* (или *нормированием*) с.в. X. Величина X/σ_X называется *нормированной* случайной величиной.

Напоминание. Пусть $M(X) = m_X$. Тогда с.в. $X- m_X$ называется *центрированной.* Математическое ожидание центрированной с.в. $M(X- m_X) = 0$.

Утверждение 2. Пусть X – с.в.; $M(X) = m_X$; $D(X) = (\sigma_X)^2$, где m_X, σ_X – числа. Пусть $Y = (X - m_X)/ \sigma_X$ – центрированная и нормированная величина для X. Тогда

$$M(X) = 0; D(X) = 1$$

Доказательство – ясно.

Утверждение 3. Пусть $X_1, …, X_n$ – n независимых случайных величин, причем математические ожидания и дисперсии у всех величин одинаковые и равны соответственно m и d. Пусть $Y = (X_1, +…+ X_n)/n$. Тогда:

$$M(Y) = m; \quad D(Y) = d/n$$

Доказательство – ясно.

8. Неравенство Чебышёва

Теорема. Пусть X - случайная величина; μ – её математическое ожидание и $D = \sigma^2$– ее дисперсия. Тогда для произвольного числа a>0 выполнено:

$$\mathbb{P}\left(|X-\mu| \geqslant a\right) \leqslant \frac{\sigma^2}{a^2},$$

Доказательство – пока отложим.

Следствие. Пусть $a = k*\sigma$. Тогда получаем:

$$\mathbb{P}\left(|X-\mu| \geqslant k\sigma\right) \leqslant \frac{1}{k^2}.$$

В частности, случайная величина с конечной дисперсией отклоняется от среднего больше, чем на 2 стандартных отклонения, с вероятностью меньше 25%. Она отклоняется от среднего на 3 стандартных отклонения с вероятностью меньше $1/9 \approx 11.2\%$.

Замечание. На занятии 4 было отступление про графы и гиперграфы. См. часть 3.

Занятие 5. 12.08 (конец занятия)

Основная часть занятия была посвящена машинному обучению. В конце был завершающий фрагмент по теории вероятностей.

9. Центральная предельная теорема

Теорема (полуформальная формулировка). Пусть X_1, …, X_n, .. – независимые одинаково распределенные случайные величины, причем математические ожидания и дисперсии у всех величин равны соответственно μ и σ^2. Пусть $S_n = X_1$, +…+ X_n, где n = 1, 2, … . Пусть $T_n = (S_n – \mu*n)/ (\sigma*sqrt(n))$ – центрированная и нормированная с.в. для S_n. Тогда T_n «имеет распределение похожее» на «нормальное распределение со средним 0 и дисперсией 1».

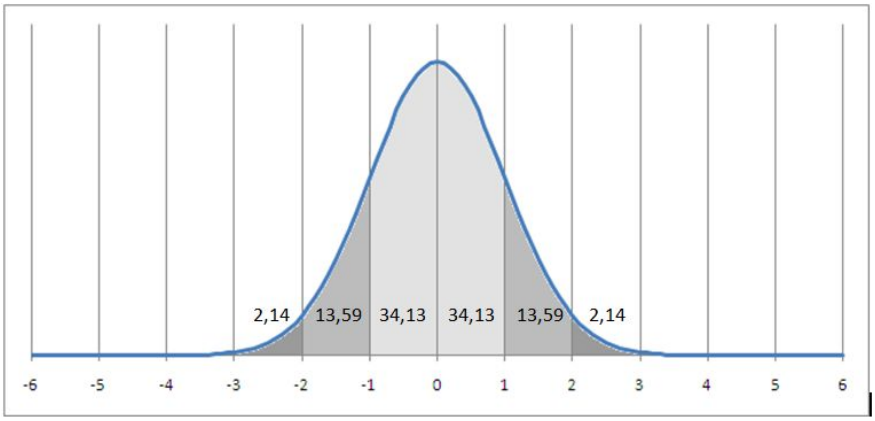


Рис.1. Нормальное распределение N(0, 1).

Часть 2.

Экосистема Python и Машинное Обучение

Занятия 5 (1-я часть), 6 и 7. 12, 13 и 14 августа.
На занятии 5 были разобраны пп. 1 – 5. Потом они были повторены на занятии 6.
Занятия 6 и 7 вёл Алексей Драль

Целью занятий было создание программы, способной предсказать цену дома по ряду его известных характеристик (такие программы мы называем *регрессорами*, см. ниже). Регрессор создавался в виде блокнота Ipython notebook; его заготовка SKLearnM изучалась во время занятия. Ниже каждый раздел текста соответствует очередной странице блокнота или части страницы.

Замечание. Термин *regressor* является общепринятым в англоязычной литературе. В русскоязычной литературе слово “регрессор” употребляется и в других смыслах. Мы здесь для краткости будем его использовать, как перевод английского “regressor”.

1. Библиотеки (см. страницу 1а).

Стандартные библиотеки, используемые при решении задач машинного обучения (Machine Learning) в Python:

- matplotlib (http://matplotlib.org/) - библиотека для рисования графиков (очень удобно для визуализации данных или результатов экспериментов)
- numpy (http://www.numpy.org/) - библиотека для эффективной работы с матрицами (одномерные - массивый, двумерные, …, многомерные)
- pandas (http://pandas.pydata.org/) - “Excel” для Python (удобно для анализа “табличных” данных, расчет базовых статистик)
- sklearn (http://scikit-learn.org/stable/) - набор алгоритмов (классификаторов, регрессов, …) для решения задач машинного обучения (например, на занятии был рассмотрен GradientBoostingRegressor - бустинг на решающих деревьях)
-

2. Наборы данных (см. страницу 1б).

Питон содержит много наборов данных (datasets). Доступ к ним возможен с помощью библиотеки dataset. У нас на занятии использовался набор boston – см. строку

```
boston = datasets.load_boston()
```

Содержание набора описано в ПРИЛОЖЕНИИ 1.

Много данных доступно по адресу http://mlcomp.org/ . Доступ к ним:

```
datasets.load_mlcomp(name_or_id)
```

Здесь name_or_id - целочисленный id или название набора данных в MLComp; список наборов данных с именами приведен на странице http://mlcomp.org/datasets.

3. Подготовка данных для проведения эксперимента (см. страницу 2).

Данные нужно (а) перемешать и (б) разделить на обучающую и тестовую выборку (train and test parts). Перемешивать нужно, чтобы избежать проблем в обучении, связанных с возможной структуризацией данных в исходном наборе данных.

Выделение тестового набора необходимо, т.к. проверять результаты обучения необходимо на наборе данных, который не пересекается с набором данных, на котором проходило обучение. В противном случае оптимальным классификатором стал бы классификатор, который просто запомнит правильные ответы. :)

Программные интерфейс:

```
X_train, X_test, y_train, y_test = train_test_split(boston.data, boston.target,
random_state=100500, train_size=0.9)
```

В данном случае у нас 90% данных используется для обучения регрессора, а 10% для оценки качества.

4. Контейнер регрессора (страница 3).

На странице 3 создается “заготовка создаваемого регрессора”. По-научному эта заготовка называется “*контейнер*”. В контейнере описывается общий алгоритм построения регрессора (в нашем случае - это алгоритм BoostingRegressor, см. ПРИЛОЖЕНИЕ 2), устанавливаются значения некоторых параметров (в нашем случае - параметры 'n_estimators', 'max_depth', 'min_samples_split', 'learning_rate', 'loss').

5. Обучение регрессора и проверка качества предсказания (страница 4).

Обучение регрессора производится с помощью функции fit, а предсказание с помощью обученного регрессора - с помощью функции predict. Эти функции существуют для любых регрессоров, поддерживаемых экосистемой Питон.

Вдаваться в детали процесса обучения мы не будем. Остановимся только на важности способа сравнения результатов. Его выбор зависит от решаемой задачи и существенно влияет на результаты обучения. В качестве примеров укажем средне-квадратичное отклонение предсказанного набора результатов от эталонного (квадрат “евклидова расстояния” между ними), а также максимальное уклонение предсказанного результата от соответствующего эталона.

В этом месте началось занятие 7

На этом занятии мы анализировали качество предсказаний.
Было много неожиданностей – совсем, как в жизни ☺

6. Обучение регрессора и проверка качества предсказания (страница 4) – эксперименты.

Выполняем страницу 4. Программа печатает значение средне-квадратичного отклонения (см. http://scikit-learn.org/stable/modules/model_evaluation.html#mean-squared-error).

Делаем несколько запусков. Оказывается, результаты отличаются.
Причина: программа обучения *fit* использует датчик случайных чисел.

7. Построение точечной диаграммы (scutter-plot), страница 5.

Программный интерфейс:

```
plt.scatter(clf.predict(X_test), y_test)
```

Результат: см. рис.1.

Каждая точка A соответствует дому. Абсцисса X_A – предсказанная цена дома; ордината Y_A – реальная цена. При хорошем качестве предсказания точки лежат вблизи диагонали. В нашем случае так оно и есть. Можно заметить, что линия

немного смещена вниз от диагонали. Это значит, что мы немного завышаем цену дома.

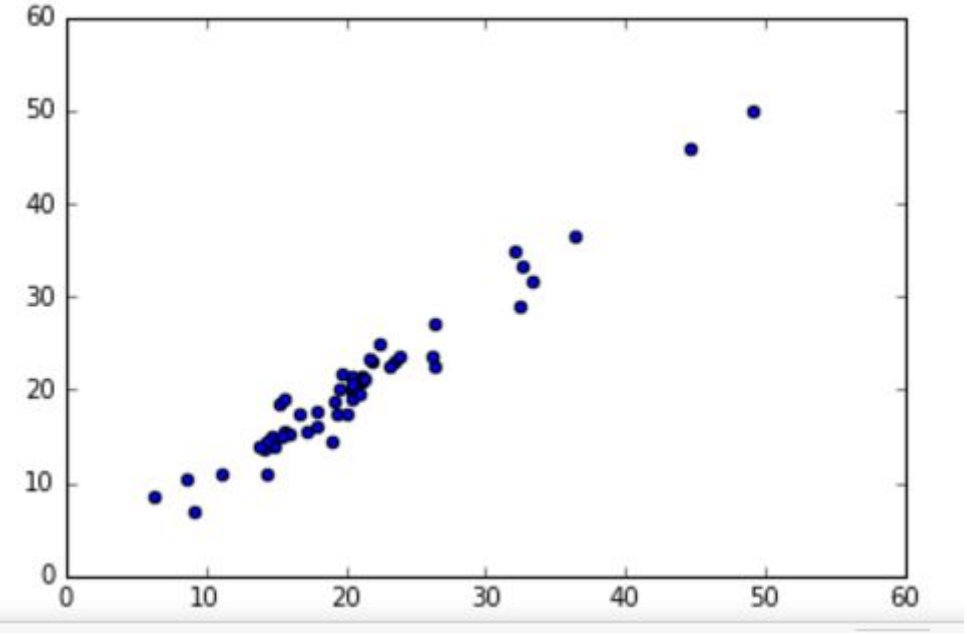
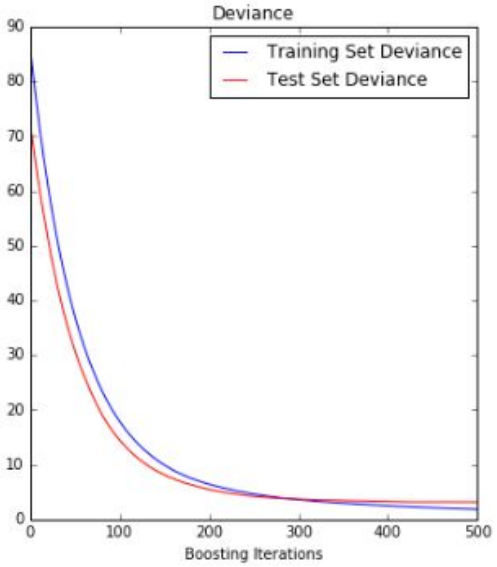


Рис.1.

8. Кривые обучения (learning curves), страница 6а

Кривые обучения полезно использовать при работе с ансамблями распознавателей (классификаторов или регрессоров); они применимы, как при работе по схеме бустинга, так и по схеме бэггинга.



Обучающая кривая метода предсказания (в нашем случае – GradientBoosting) для данного набора данных – это набор точек вида (R, Q), где R – количество распознавателей в ансамбле; Q – качество предсказания при R распознавателях (остальные параметры метода, в том числе способ оценки качества предсказания при этом фиксированы). Обычно строят две кривые – для тренировочного и тестового набора данных. То, что получилось у нас – см. рис.2.

Обратите внимание: тестовая кривая начиная примерно с 300 распознавателей находится на одном уровне, а тренировочная продолжает убывать.

Рис. 2

Этот эффект называется ***переобучение.*** Дело в том, что, имея много «свободных» параметров, можно достичь практически идеального качества предсказания на обучающей выборке. Вот только при работе с внешними данными это пользы не принесет. Поэтому всегда нужно проверять работу распознавателя с помощью кривых обучения.

ВНИМАНИЕ! С этим графиком ЧТО-ТО НЕ ТАК!!!

Качество обучения на тестовой выборке (судя по рис.2) лучше, чем на обучающей выборке. Это противоречит здравому смыслу. С такими ситуациями обязательно нужно разбираться, прежде, чем переходить к практическому использованию метода.

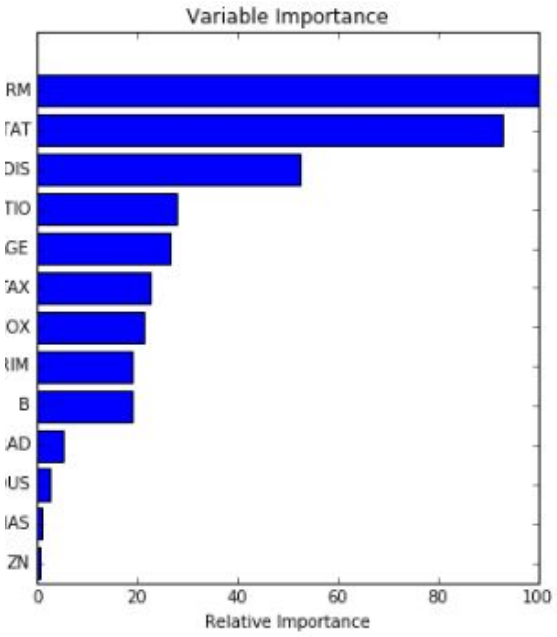
Первое, что нужно сделать – попробовать, сохраняется ли эффект при изменении разбиения на тестовую и обучающую выборку (см. п.3). Мы попробовали – все сохранилось.

Значит, дело, скорее всего в программе. Алексей (после занятия) начал разбираться, нашел интересные вещи. Но ясности пока нет. Будет – сообщим.

8. Таблица значимости признаков, страница 6б

В пакете SKLearn есть программа, которая определяет значимости признаков для распознавателя. Интерфейс:

feature_importance = clf.feature_importances_



Результат – список значимостей признаков (точнее – это не список, а словарь – индексами в нем являются признаки, а не натуральные числа).

Что означают эти значимости, сейчас разбираться не будем. Важно понимать: чем больше, тем лучше. При изображении значимости удобно разде-лить на максимальное значение – после этого они будут в диапазоне о 0 до 1.

То, что у нас получилось, изображено на рис.3. К счастью, все согласуется со здравым смыслом.

Рис.3.

9. Подбор значений параметров, страница 7.

На стр.3 мы установили значения параметров «из общих соображений». После того, как мы убедились, что «в целом» все работает, стоит проверить другие значения параметров. Например, попробуем значения параметров "max_depth" и "min_samples_split" вот в таких диапазонах:

```
"max_depth": range(1,11),
"min_samples_split": [1, 2, 5, 10]
```

Можно было бы написать перебор всех пар значений в цикле. Но в SKLearn есть специальная функция для перебора параметров *GridSearchCV.*
(см. http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.GridSearchCV.html)

Мы воспользовались этой функцией. Результат – таблица значений погрешностей и параметры, соответствующие наименьшей погрешности.

ВНИМАНИЕ! Снова расхождение со здравым смыслом: На стр. 4 значение погрешности примерно 3.11, а в таблице – значительно выше. Возможная причина - неоднородность набора данных «Бостон» (повезет – будет хорошее качество предсказания, не повезет – будет плохое). В реальной жизни – нужно тщательно разбираться.

10. Что еще было.

В конце занятия бегло попробовали предсказывать стоимость 1Гб внешней памяти. Детали здесь разбирать не будем.

Основной смысл:

1) можно не только разбираться с новыми данными, но и предсказывать будущее;
2) как и всегда, нужно тщательно оценивать качество предсказаний.

ПРИЛОЖЕНИЕ 1

Описание набра данных “Бостон”

 # print(boston.DESCR)

Boston House Prices dataset

Notes

Data Set Characteristics:

:Number of Instances: 506

:Number of Attributes: 13 numeric/categorical predictive

:Median Value (attribute 14) is usually the target

:Attribute Information (in order):

- CRIM per capita crime rate by town
- ZN proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS proportion of non-retail business acres per town
- CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX nitric oxides concentration (parts per 10 million)
- RM average number of rooms per dwelling
- AGE proportion of owner-occupied units built prior to 1940
- DIS weighted distances to five Boston employment centres
- RAD index of accessibility to radial highways
- TAX full-value property-tax rate per $10,000
- PTRATIO pupil-teacher ratio by town
- B 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT % lower status of the population
- MEDV Median value of owner-occupied homes in $1000's

:Missing Attribute Values: None

:Creator: Harrison, D. and Rubinfeld, D.L. This is a copy of UCI ML housing dataset.

http://archive.ics.uci.edu/ml/datasets/Housing

This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University.

The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978. Used in Belsley, Kuh & Welsch, 'Regression diagnostics ...', Wiley, 1980. N.B. Various transformations are used in the table on pages 244-261 of the latter.

The Boston house-price data has been used in many machine learning papers that address regression problems.

References

- Belsley, Kuh & Welsch, 'Regression diagnostics: Identifying Influential Data and Sources of Collinearity', Wiley, 1980. 244-261.
- Quinlan,R. (1993). Combining Instance-Based and Model-Based Learning. In Proceedings on the Tenth International Conference of Machine Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.
- many more! (see http://archive.ics.uci.edu/ml/datasets/Housing)

ПРИЛОЖЕНИЕ 2

Алгоритм GradientBoosting

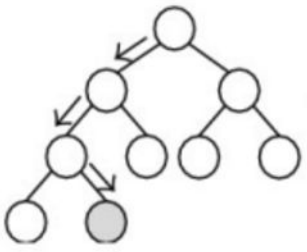
Алгоритм GradientBoosting состоит в использовании ансамбля классификаторов “Дерево” (см. ПРИЛОЖЕНИИ 3), взаимодействующих по методу “Бустинг” (см. ПРИЛОЖЕНИЕ 4). Слово “Gradient” указывает на особенность обучения классификатора, здесь мы его объяснять не будем. Параметры, указанные в скрипте, имеют следующий смысл: 'n_estimators' - количество классификаторов в ансамбле; 'max_depth' - максимальная глубина дерева; 'min_samples_split' - минимальное количество элементов обучающей выборки, приписанное листу дерева-классификатора; 'learning_rate' - связан с особенностями метода обучения; 'loss'- описывает способ вычисления погрешности предсказания (см. выше п.5).

ПРИЛОЖЕНИЕ 3

Классификатор “Дерево”

Классификатор “Дерево” (по-английски tree classifier) устроен следующим образом. Во время обучения каждому внутреннему узлу дерева сопоставляется условие на один из признаков, а каждому листу - множество объектов обучающей выборки. По этому множеству строится доля представителей каждого класса (для задачи классификации) или среднее значение целевого признака (для задачи регрессии).

При анализе какого-либо объекта мы перемещаемся от корня дерева к листьям в соответствии со значениями признаков объекта. Ответ выдается в соответствии с достигнутым листом.

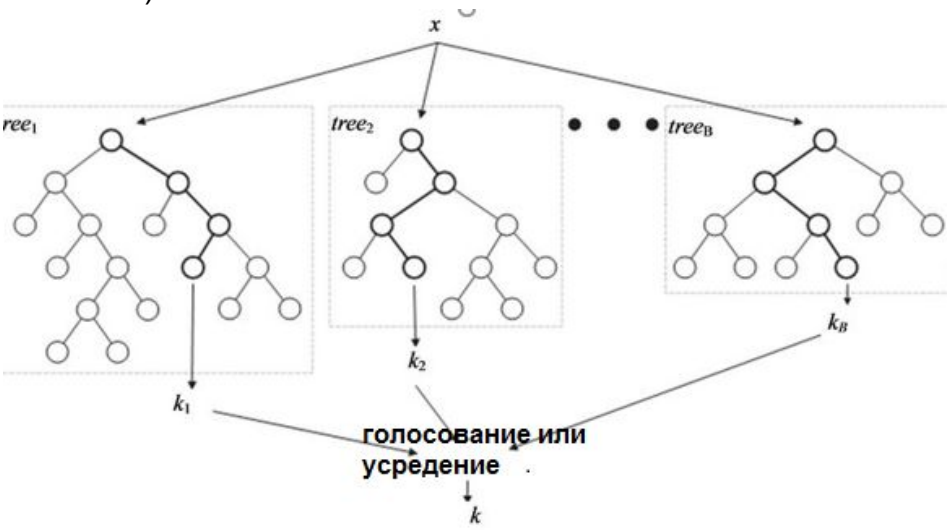


ПРИЛОЖЕНИЕ 4

Ансамбли классификаторов. Bagging and Boosting

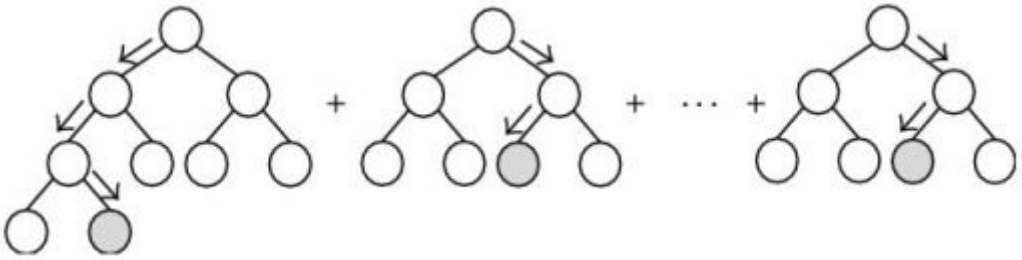
Есть два стандартных подхода для использования ансамблей классификаторов/регрессоров: bagging (бэггинг) и boosting (бустинг).

На занятии мы не говорили слово “бэггинг”, но фактически мы его обсудили, когда говорили про RandomForest (он же случайный лес с небольшими оговорками, которые мы здесь опустим для простоты изложения). При использовании бэггинга все классификаторы ансамбля решают одну и ту же задачу и работают независимо. Итоговый результат получается путем «совмещения» их результатов (см. рисунок; здесь классификаторы - это деревья, что не обязательно)



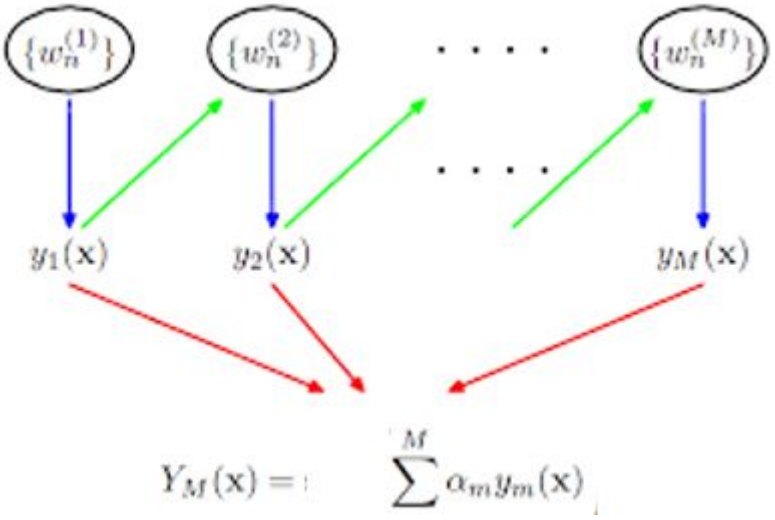
Сводный результат в случае классификаторов получается голосованием; в случае регрессоров – осреднением. Идея бэггинга в том, что каждый из классификаторов ошибается на разных входных данных и ошибается «в разные стороны». Поэтому усреднение по ответам всех классификаторов обычно дает (в целом по всем входам) лучший результат, чем каждый классификатор по отдельности.

Бустинг может применяться как для задач классификации, так и для задач регрессии. Здесь мы рассматриваем только применение бустинга при решении задачи регрессии, т.е. когда результатом предсказания является число.



При бустинге классификаторы работают последовательно; будем считать их перенумерованными – нулевой, первый и т.д. Нулевой классификатор обучен решать исходную задачу. Следующий (первый) классификатор предсказывает

погрешность результата первого классификатора (разность Y1 = Y0 - P0 между эталонным результатом Y и предсказанием P0 нулевого классификатора). Пусть P1 - это предсказание 1-го классификатора, а Y2 = Y1 - P1 = Y0-P0-P1 - его погрешность. Классификатор №2 предсказывает погрешность 1-го классификатора - величину Y2. Пусть P2 - это предсказание классификатора №2. Классификатор №3 предсказывает погрешность Y3 = Y2-P2 = Y0-P0-P1-P2 классификатора №2. И т.д. Окончательное предсказание значения величины Y0 получается, таким образом, как сумма предсказаний всех классификаторов P0+P1+P2+... .



Этот принцип работы ансамбля классификаторов можно назвать принципом “Репки”. Мы обучаем первый классификатор (“дедка”) и получаем какое-то предсказание. Оно неточное, потому подключается “бабка”, которая помогает исправить ошибки предсказания. В сумме, они тоже ошибаются, поэтому подключается “внучка” и так далее.

Часть 3.

Скрытые марковские модели

См. презентацию.