Анализ опроса словацкой молодежи 2013

Никита Козловский

ВГУ, ПММ, 3 группа

Содержание

1 Введение

Рассмотрим данные, представленные на сайте https://www.kaggle.com/miroslavsabo/young-people-survey Опрос был проведен факультетом факультет социальных и экономических наук Университета имени Коменского в Братиславе в 2013 году. Было опрошено 1010 человек по 150 пунктам:

- Предпочтения в музыке (19 пунктов)
- Предпочтения в фильмах (12 пунктов)
- Хобби и интересы (32 пунктов)
- Фобии (10 пунктов)
- Здравоохранение (3 пунктов)
- Черты характера, взгляды на жизнь и мнения (57 пунктов)
- На что тратите деньги (7 пунктов)
- Демографические данные (10 пунктов)

Попробуем найти ответы на следующие вопросы:

- Можно ли разделить студентов на группы по предпочтениям в музыке? (описательные, факторный)
- Что определяет бережливого человека ? (регрессия)
- Анализ страхов по группам

Пример данных

```
In [24]: df.head(4)
```

Out[24]:

	Music	Slow songs or fast songs	Dance	Folk	Country	Classical music	\
0	5.0	3.0	2.0	1.0	2.0	2.0	
1	4.0	4.0	2.0	1.0	1.0	1.0	
2	5.0	5.0	2.0	2.0	3.0	4.0	
3	5.0	3.0	2.0	1.0	1.0	1.0	

	Musical	Pop	Rock	Metal or Hardrock	...	Age	\
0	1.0	5.0	5.0	1.0	...	20.0	
1	2.0	3.0	5.0	4.0	...	19.0	
2	5.0	3.0	5.0	3.0	...	20.0	
3	1.0	2.0	2.0	1.0	...	22.0	

	Height	Weight	Number of siblings	Gender	Left - right handed	\
0	163.0	48.0	1.0	female	right handed	
1	163.0	58.0	2.0	female	right handed	
2	176.0	67.0	2.0	female	right handed	
3	172.0	59.0	1.0	female	right handed	

```
                 Education  Only child  Village - town  House - block of flats
0  college/bachelor degree          no         village          block of flats
1  college/bachelor degree          no            city          block of flats
2         secondary school          no            city          block of flats
3  college/bachelor degree         yes            city          house/bungalow

[4 rows x 150 columns]
```

2 Анкета

2.1 Музыкальные предпочтения

1. Мне нравится слушать музыку: Категорически не согласен 1-2-3-4-5 Полностью согласен (целое)
2. Я предпочитаю: Медленную музыку 1-2-3-4-5 Быструю музыку (целое)
3. Dance, Disco, Funk: Не нравится вообще1-2-3-4-5 Нравится очень (целое)
4. Народная музыка: Не нравится вообще 1-2-3-4-5 люблю (целое)
5. Кантри: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
6. Классика: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
7. Мюзиклы: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
8. Поп: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
9. Рок: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
10. Металл, хард-рок: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
11. Панк: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
12. Хип-хоп, Рэп: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
13. Reggae, Ska: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
14. Swing, Jazz: не нравится вообще 1-2-3-4-5 Нравится очень (целое)
15. Рок-н-ролл: не нравится вообще 1-2-3-4-5 Нравится очень (целое)
16. Альтернативная музыка: не нравится вообще 1-2-3-4-5 Нравится очень (целое)
17. Латино: не нравится вообще 1-2-3-4-5 Нравится очень (целое)
18. Техно, Транс: не нравится вообще 1-2-3-4-5 Нравится очень (целое)
19. Опера: не нравится вообще 1-2-3-4-5 Нравится очень (целое)

2.2 Предпочтения в фильмах

1. Мне очень нравится смотреть фильмы: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
2. Фильмы ужасов: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
3. Триллеры: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
4. Комедии: не нравится вообще 1-2-3-4-5 Нравится очень (целое)
5. Романтические фильмы: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
6. Научно-фантастические фильмы: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)

7. Военные фильмы: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
8. Сказки: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
9. Мультфильмы: Не нравится вообще 1-2-3-4-5 Наслаждайтесь очень (целое)
10. Документальные фильмы: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
11. Западные фильмы: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)
12. Экшн фильмы: Не нравится вообще 1-2-3-4-5 Нравится очень (целое)

2.3 Интересы и хобби

1. История: Не интересует 1-2-3-4-5 Очень интересно (целое)
2. Психология: не интересует 1-2-3-4-5 Очень интересно (целое)
3. Политика: Не интересует 1-2-3-4-5 Очень интересно (целое)
4. Математика: не интересует 1-2-3-4-5 Очень интересно (целое)
5. Физика: не интересует 1-2-3-4-5 Очень интересно (целое)
6. Интернет: не интересует 1-2-3-4-5 Очень интересно (целое)
7. Программное обеспечение для ПК, Оборудование: Не интересует 1-2-3-4-5 Очень интересно (целое)
8. Экономика, Менеджмент: Не интересует 1-2-3-4-5 Очень интересно (целое)
9. Биология: не интересует 1-2-3-4-5 Очень интересно (целое)
10. Химия: не интересует 1-2-3-4-5 Очень интересно (целое)
11. Чтение стихов: Не интересует 1-2-3-4-5 Очень интересно (целое)
12. География: Не интересует 1-2-3-4-5 Очень интересно (целое)
13. Иностранные языки: не интересуются 1-2-3-4-5 Очень интересно (целое)
14. Медицина: не интересует 1-2-3-4-5 Очень интересно (целое)
15. Закон: не интересует 1-2-3-4-5 Очень интересно (целое)
16. Автомобили: Не интересует 1-2-3-4-5 Очень интересно (целое)
17. Искусство: Не интересует 1-2-3-4-5 Очень интересно (целое)
18. Религия: Не интересует 1-2-3-4-5 Очень интересно (целое)
19. Мероприятия на свежем воздухе: не интересует 1-2-3-4-5 Очень интересно (целое)
20. Танцы: не интересуются 1-2-3-4-5 Очень интересно (целое)
21. Игра на музыкальных инструментах: не интересует 1-2-3-4-5 Очень интересно (целое)

22. Поэзия: Не интересно 1-2-3-4-5 Очень интересно (целое)
23. Занятия спортом: Не интересуюсь 1-2-3-4-5 Очень интересно (целое)
24. Спорт на конкурентном уровне: не интересует 1-2-3-4-5 Очень интересно (целое)
25. Садоводство: Не интересует 1-2-3-4-5 Очень интересно (целое)
26. Знаменитый образ жизни: не интересует 1-2-3-4-5 Очень интересно (целое)
27. Покупки: Не интересует 1-2-3-4-5 Очень интересно (целое)
28. Наука и техника: не интересует 1-2-3-4-5 Очень интересно (целое)
29. Театр: не интересует 1-2-3-4-5 Очень интересно (целое)
30. Общение: Не интересует 1-2-3-4-5 Очень интересно (целое)
31. Адреналин спорт: не интересует 1-2-3-4-5 Очень интересно (целое)
32. Домашние животные: Не интересуются 1-2-3-4-5 Очень интересно (целое)

2.4 Фобии

1. Полет: совсем не боюсь 1-2-3-4-5 Очень боюсь (целое)
2. Гром, молния: совсем не боюсь 1-2-3-4-5 Очень боюсь (целое)
3. Тьма: совсем не боюсь 1-2-3-4-5 Очень боюсь (целое)
4. Высота: совсем не боюсь 1-2-3-4-5 Очень боюсь (целое)
5. Пауки: совсем не боюсь 1-2-3-4-5 Очень боюсь (целое)
6. Змеи: совсем не боюсь 1-2-3-4-5 Очень боюсь (целое)
7. Крысы, мыши: совсем не боюсь 1-2-3-4-5 Очень боюсь (целое)

2.5 Отношение к здоровью

1. Отношение к курению: Никогда не курил - Пробовал курить - Бывший курильщик - Текущий курильщик (Номинальная)
2. Пью: Никогда - Пью в компаниях - Пью много (Номинальная)
3. Я живу очень здоровым образом жизни: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

2.6 Черты характера, взгляды на жизнь

1. Я обращаю внимание на то, что происходит вокруг: сильно не согласен 1-2-3-4-5. Полностью согласен (целое)
2. Я стараюсь выполнять задания как можно скорее и не оставлять их до последней минуты .: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
3. Я всегда составляю список, поэтому ничего не забываю: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
4. Я часто учусь или работаю даже в свободное время .: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
5. Я смотрю на вещи с разных точек зрения, прежде чем идти вперед. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)
6. Я считаю, что плохие люди пострадают в один прекрасный день, и хорошие люди будут вознаграждены. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)
7. Я надёжен на работе и всегда выполняю все заданные мне задачи: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)
8. Я всегда соблюдаю свои обещания: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
9. Я могу быстро попасть на кого-то, а затем полностью потерять интерес. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)
10. Я бы предпочел иметь много друзей, чем много денег. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)
11. Я всегда стараюсь быть самым смешным: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
12. Иногда я могу столкнуться с двумя лицами: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)
13. Я повредил вещи в прошлом, когда сердился .: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
14. Я не тороплюсь, чтобы принимать решения: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
15. Я всегда стараюсь голосовать на выборах: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
16. Я часто думаю и сожалею о принимаемых мной решениях: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)
17. Я могу сказать, слушают ли меня люди или нет, когда я говорю с ними: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)
18. Я ипохондрик. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)

19. Я - эмфатичный человек. Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

20. Я ем, потому что должен. Я не наслаждаюсь едой и еду так быстро, как могу: Категорически не согласен 1-2-3-4-5 Полностью согласен (целое)

21. Я стараюсь отдать столько, сколько я могу, другим людям на Рождество: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

22. Мне не нравится встречаться с животными: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

23. Я ухаживаю за вещами, которые я заимствовал у других: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

24. Я чувствую себя одиноким в жизни: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

25. Раньше я учился в школе: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

26. Я беспокоюсь о своем здоровье: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

27. Хотел бы я изменить прошлое из-за того, что я сделал: Полностью не согласен 1-2-3-4-5 Сильно соглашаюсь (целое)

28. Я верю в Бога: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

29. У меня всегда хорошие мечты: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

30. Я всегда отдаю благотворительность. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)

31. У меня много друзей: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

32. Сроки: Я часто бываю на ранней стадии. - Я всегда вовремя. - Я часто опаздываю. (Категорично)

33. Вы лжете другим: Никогда. - Только чтобы не причинять кому-либо вреда. Иногда. - Каждый раз мне это подходит. (Номинальная)

34. Я очень терпелив: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

35. Я могу быстро адаптироваться к новой среде .: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

36. Мое настроение меняется быстро: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

37. Я хорошо воспитан, и я ухаживаю за внешностью: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

38. Мне нравится встречаться с новыми людьми .: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

39. Я всегда позволяю другим людям узнать о моих достижениях: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

40. Я думаю, тщательно, прежде чем отвечать на любые важные письма .: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

41. Я люблю детей: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

42. Я не боюсь высказать свое мнение, если я сильно против чего либо: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

43. Я могу рассердиться очень легко: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

44. Я всегда убеждаюсь, что я общаюсь с нужными людьми: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

45. Я должен быть хорошо подготовлен перед публичным выступлением: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

46. Я виноват в том, люди меня не любят. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)

47. Я плачу, когда чувствую себя, или все идет не так. Абсолютно не согласен 1-2-3-4-5. Полностью согласен (целое)

48. Я на 100% доволен своей жизнью. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)

49. Я всегда полна жизни и энергии: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

50. Я предпочитаю крупных опасных собак более мелким, более спокойным собакам: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

51. Я считаю, что все мои черты характера положительные: сильно не согласны 1-2-3-4-5. Полностью согласен (целое)

52. Если я найду что-то, что не принадлежит мне, я передам его. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)

53. Мне очень трудно вставать утром: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

54. У меня много разных увлечений и интересов: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

55. Я всегда слушаю совет моих родителей: Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)

56. Мне нравится участвовать в опросах: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)

57. Сколько времени вы проводите в Интернете ?: Не провожу - Менее часа в день - Несколько часов в день - Большая часть дня (Номинальная)

2.7 Отношение к деньгам

1. Я могу сэкономить все, что смогу. Полностью не согласен 1-2-3-4-5. Полностью согласен (целое)
2. Мне нравится ходить в крупные торговые центры .: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
3. Я предпочитаю фирменную одежду: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
4. Я трачу много денег на вечеринки и общение: Сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
5. Я трачу много денег на внешний вид: сильно не согласен 1-2-3-4-5 Полностью согласен (целое)
6. Я трачу много денег на гаджеты: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)
7. Я буду с удовольствием платить больше денег за хорошее, качественное или здоровое питание: Полностью не согласен 1-2-3-4-5 Полностью согласен (целое)

2.8 Данные о себе

1. Возраст: (целое)
2. Высота: (целое)
3. Вес: (целое)
4. Сколько у вас братьев и сестер ?: (целое)
5. Пол: Женский - Мужской (Номинальная)
6. Я: Левая рука - Правая (Номинальная)
7. Высшее образование: В настоящее время ученик начальной школы - Начальная школа - Средняя школа - Колледж / Степень бакалавра (Номинальная)
8. Я единственный ребенок: Нет - Да (Номинальная)
9. Я провел большую часть своего детства в: Городе - деревне (Номинальная)
10. Я прожил большую часть своего детства в: доме - многоквартирном доме (Номинальная)

3 Описательные статистики

3.1 Основные численные характеристики

Из-за большого количества признаков, рассмотрим лишь подможество из множества ответов, а именно рассмотрим предпочтения в музыке. Для каждой характериски расчитаем

- mean – средняя величина из исходных значений $\bar{x} = \frac{x_1+x_2+\cdots+x_n}{n}$;
- std – стандартное отклонение, мера того, насколько широко разбросаны точки данных относительно их средних $s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2}$
- min, max – минимальное и максимальное значение
- 25% , 50% , 75% квантили соответствующих уровней. Значение, которое заданная случайная величина не превышает с фиксированной вероятностью. Верхняя квантиль включает 25% наибольших чисел в наборе, нижняя, соответственно, 25% наименьших чисел в наборе, 50% соответсвует медиане выборки
- skewness – коэффициент асимметрии, который показывает, насколько симметрично распределение данной случайной величины. Если коэффициент асимметрии положителен, то отклонение происходит в сторону положительных значений, в ином случае – отрицательных
- kurtosis – эксцесс – мера крутости кривой распределения. Кривая распределения может быть островершинной, плосковершинной, средне вершинной. Эти четыре момента составляют набор особенностей распределения при анализе данных. Для нормального распределения A=0, E=0. Положительный эксцесс обозначает относительно остроконечное распределение. Отрицательный эксцесс обозначает относительно сглаженное распределение
- mode – мода, наиболее часто встречаемое значение
- NAs – количество пропущенных данных.

```
In [11]: import pandas as pd
         import numpy as np
```

```
In [4]: from pandas import Series
        def customDescribe(x):
            data = [x.mean(), x.std(), x.min(), x.quantile(0.25), x.median(),
                    x.quantile(0.75), x.max(), x.skew(), x.kurtosis(), x.mode().max(),
                    x.isnull().sum()]
            names = ['mean', 'std', 'min', '25%', '50%', '75%', 'max',
             'skewness', 'kurtosis', 'mode', 'NAs']
            return Series(data, index=names)

        names = pd.read_csv('columns.csv')
        df = pd.read_csv('responses.csv')
        music = df.iloc[:,2:18]
        music.apply(customDescribe)
```

Out[4]:

	Dance	Folk	Country	Classical music	Musical	Pop
mean	3.113320	2.288557	2.123383	2.956132	2.761905	3.471698
std	1.170568	1.138916	1.076136	1.252570	1.260845	1.161400
min	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
25%	2.000000	1.000000	1.000000	2.000000	2.000000	3.000000
50%	3.000000	2.000000	2.000000	3.000000	3.000000	4.000000
75%	4.000000	3.000000	3.000000	4.000000	4.000000	4.000000
max	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000
skewness	-0.045760	0.694783	0.795798	0.107357	0.219951	-0.383317
kurtosis	-0.803331	-0.216416	-0.037576	-0.969287	-0.928080	-0.704309
mode	3.000000	2.000000	2.000000	3.000000	3.000000	4.000000
NAs	4.000000	5.000000	5.000000	7.000000	2.000000	3.000000

	Rock	Metal or Hardrock	Punk	Hiphop, Rap	Reggae, Ska
mean	3.761952	2.361470	2.456088	2.910537	2.769691
std	1.184861	1.372995	1.301105	1.375677	1.214434
min	1.000000	1.000000	1.000000	1.000000	1.000000
25%	3.000000	1.000000	1.000000	2.000000	2.000000
50%	4.000000	2.000000	2.000000	3.000000	3.000000
75%	5.000000	3.000000	3.000000	4.000000	4.000000
max	5.000000	5.000000	5.000000	5.000000	5.000000
skewness	-0.702586	0.604915	0.441427	0.037217	0.156497
kurtosis	-0.419187	-0.934732	-0.959379	-1.250059	-0.900509
mode	5.000000	1.000000	1.000000	4.000000	3.000000
NAs	6.000000	3.000000	8.000000	4.000000	7.000000

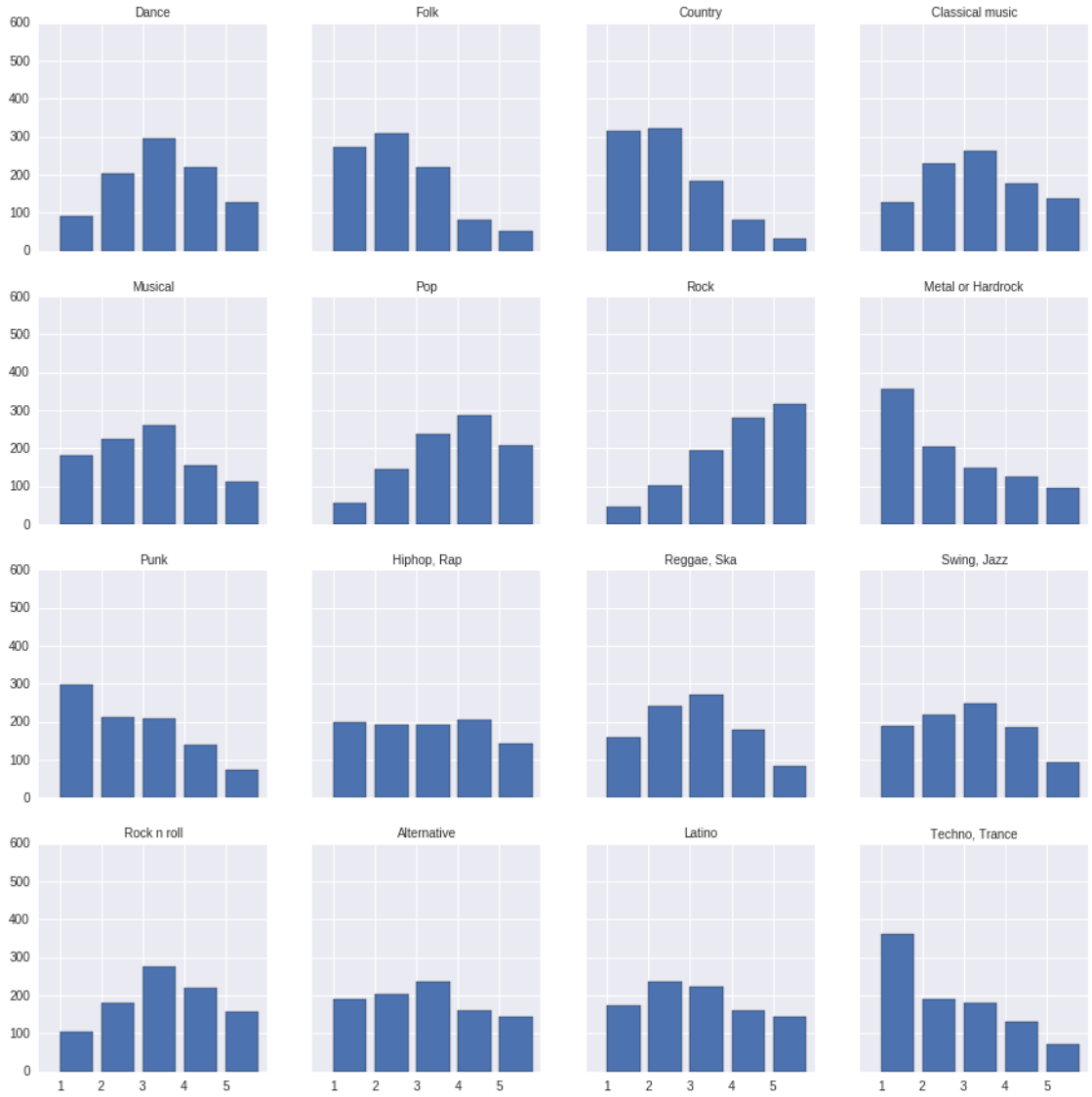
	Swing, Jazz	Rock n roll	Alternative	Latino	Techno, Trance
mean	2.759960	3.141575	2.828514	2.842315	2.338983
std	1.257936	1.237269	1.347173	1.327902	1.324099
min	1.000000	1.000000	1.000000	1.000000	1.000000
25%	2.000000	2.000000	2.000000	2.000000	1.000000
50%	3.000000	3.000000	3.000000	3.000000	2.000000
75%	4.000000	4.000000	4.000000	4.000000	3.000000
max	5.000000	5.000000	5.000000	5.000000	5.000000
skewness	0.146457	-0.108936	0.162211	0.188489	0.569644
kurtosis	-0.997739	-0.917436	-1.129404	-1.099347	-0.906037
mode	3.000000	3.000000	3.000000	2.000000	1.000000
NAs	6.000000	7.000000	7.000000	8.000000	7.000000

3.2 Визуализация данных

In [5]:
```
music = music.dropna()
```

In [9]:
```
import seaborn as sns
import matplotlib.pyplot as plt
plt_dict = {}

for i in range(0,len(music.columns)):
```

```
    plt_dict.update({i:music.columns[i]})

fig, ax = plt.subplots(4,4,figsize=(15,15), sharey=True, sharex=True)
x = [1,2,3,4,5]
initial = 0

for i in range(4):
    for j in range(4):
        y = music[plt_dict[initial]].value_counts().to_dict()
        ax[i,j].bar(y.keys(), y.values())
        ax[i,j].set_ylabel('')
        ax[i,j].set_xlabel('')
        ax[i,j].set_xticklabels(labels=np.arange(0,6), fontsize=10)
        ax[i,j].set_yticklabels(labels=np.arange(0,601,100), fontsize=10)
        ax[i,j].set_title(plt_dict[initial], fontsize=10)
        ax[i,j].set_xlim(.5,6)
        ax[i,j].set_ylim(0,600)
        initial += 1
plt.show()
```



3.3 Вывод

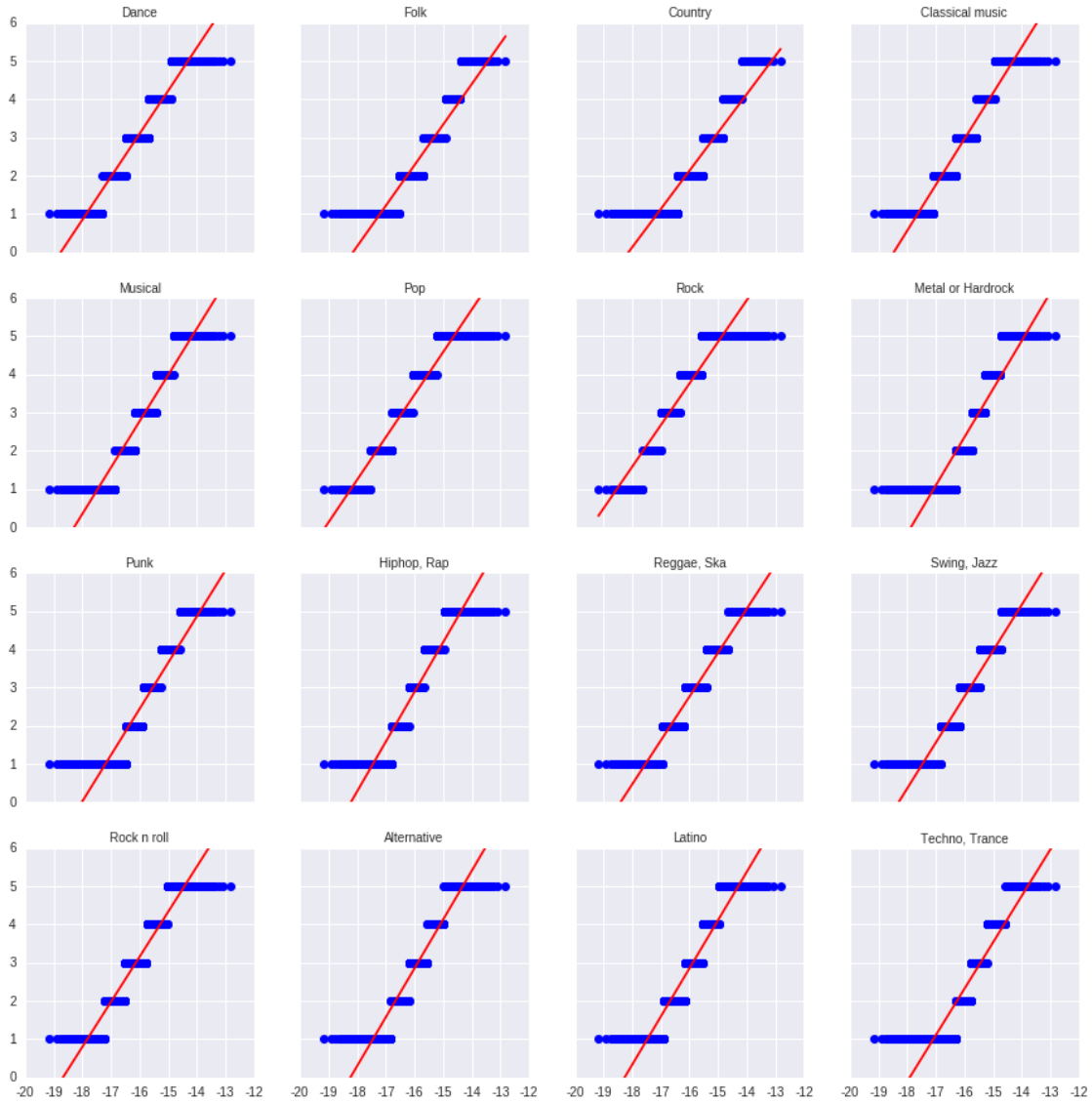
Как видно, кривые распределений плосковерхие, при этом для большинсва графиков четко прослеживается тяготение к одному из крайних допустимых значений. Это связано, во-первых, с тем, что данные носят дискретный характер, а во-вторых, сам характер данных (предпочтение в музыке) носят всеобщий и стихийный характер. Стиль или более-менее устраивает всех – альтернативная музыка, теряет популярность, как, например фолк, имеет преданных фанатав, как рок, а может вызывать неприязнь, как техно.

4 Проверка нормальности распределения

Построим normal probability plot для всех характеристик наших данных (ожидаемое и представленное значение). Как видно,данные распределены не нормально. Это же следует из предо-

ставленного окружением scipy теста normaltest, который, в частности возвращает p-value, которое показывает вероятность отклонения нулевой гипотезы, которая состоит в том, что данные распределены равномерно.

```
In [38]: from scipy.stats.mstats import normaltest
         from scipy.stats import probplot
         import seaborn as sns
         import matplotlib.pyplot as plt
         plt_dict = {}

         for i in range(0,len(music.columns)):
             plt_dict.update({i:music.columns[i]})

         fig, ax = plt.subplots(4,4,figsize=(15,15), sharey=True, sharex=True)
         x = [1,2,3,4,5]
         initial = 0

         for i in range(4):
             for j in range(4):
                 probplot(music[plt_dict[initial]], plot=ax[i,j])
                 ax[i,j].set_ylabel('')
                 ax[i,j].set_xlabel('')
                 ax[i,j].set_xticklabels(labels=np.arange(-20,20), fontsize=10)
                 ax[i,j].set_yticklabels(labels=np.arange(0,10,1), fontsize=10)
                 ax[i,j].set_title(plt_dict[initial], fontsize=10)
                 ax[i,j].set_ylim(0,6)

                 initial += 1
         plt.show()

         normaltest(music).pvalue
```




```
Out[38]: array([  1.73933578e-018,   9.70589103e-015,   9.08852599e-018,
                  3.03938325e-039,   1.85007144e-036,   1.81645640e-017,
                  1.15204030e-016,   6.09948269e-042,   3.13553153e-039,
                  2.84113677e-210,   7.36973614e-024,   2.01589772e-045,
                  2.74815258e-029,   1.90506599e-087,   1.42014654e-076,
                  2.82482142e-037])
```

5 Факторный анализ

5.1 Постановка задачи

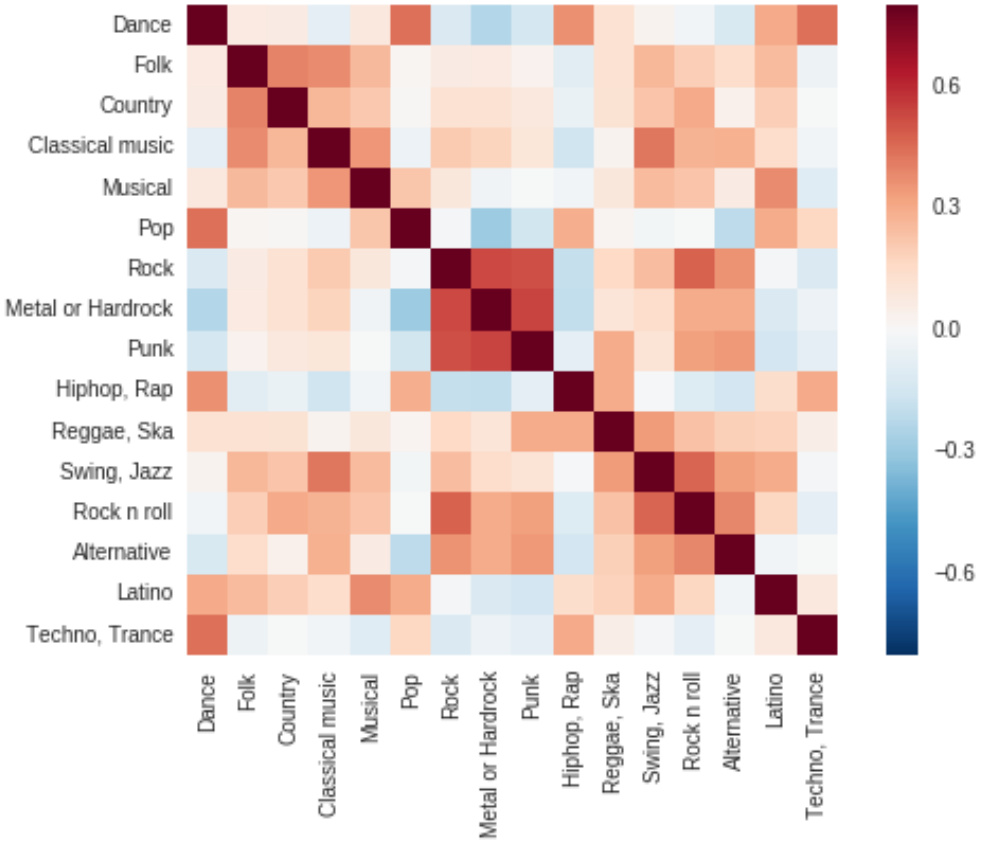
Попробуем уменьшить предпочтения в музыке в несколько удобных, легко интерпретируемых факторов, которые лучше помогут понять ответы респондендентов. Подобный подход может

быть применен по любой другой схожей категории ответов (по фильмам, хобби и т.д).

5.2 Корреляции

Для начала посмотрим, как данные коррелируют друг с другом, воспользовавшись коэфициентом Пирсона $\mathbf{r}_{XY} = \frac{\mathbf{cov}_{XY}}{\mathbf{\alpha}_X \sigma_Y} = \frac{\sum(X-\bar{X})(Y-\bar{Y})}{\sqrt{\sum(X-\bar{X})^2}\sqrt{\sum(Y-\bar{Y})^2}}$

```
In [15]: import seaborn as sbn
         import matplotlib.pyplot as plt
         corr = music.corr()
         sbn.heatmap(corr,
                     xticklabels=corr.columns.values,
                     yticklabels=corr.columns.values,
                     vmax=.8, square=True)
         plt.show()
```



Данные связанны между собой довольно очевидным образом, например классическая музыка и опера, рок, панк и тяжелый метал, а так же наблюдаются другие, не менее очевидные корреляции.

5.3 Описание метода

Проведем факторый анализ. Он имеет простую линейную форму

$$z_{ai} = \sum_p \ell_{ap} F_{pi} + \varepsilon_{ai},$$

где F_{pi} это общие факторы, а ε_{ai} характерный фактор i-ого показателя. Дисперсия любого исходного показателя состоит из общности и характерности. Общность характеризует ту часть дисперсии исходного показателя, которая объясняется общими факторами. Ставится задача по исходным показателям подобрать так общие факторы, чтобы они как можно в большей мере вариацию исходных показателей. В модели делается предположение, что исходные показатели имеют среднее значение, равное 0 и дисперсию = 1. Это предположение не выполняется ни для одной переменной, поэтому предварительно данные нормируются.

Окружение scikit learn предоставляет програмные средства для подбора оптимального числа факторов, а так же обучаемую модель.

```
In [69]: import numpy as np
         import matplotlib
         import matplotlib.pyplot as plt
         from sklearn.model_selection import cross_val_score

         n_features = len(music.columns)
         n_components = np.arange(0,  n_features)
         fa_scores = []
         fa = FactorAnalysis()
         fa.fit(music)
         for n in n_components:
             fa.n_components = n
             fa_scores.append(np.mean(cross_val_score(fa, music)))
         n_components_fa = n_components[np.argmax(fa_scores)]
         print(n_components_fa)
```

```
3
```

```
In [72]: from sklearn.decomposition import FactorAnalysis
         factor = FactorAnalysis(n_components=3)
         factor.fit(music)
         print ((pd.DataFrame(factor.components_,columns=music.columns)).transpose())
```

```
                          0         1         2
Dance             -0.303149  0.665696 -0.425720
Folk               0.311782  0.402950  0.319220
Country            0.313558  0.329835  0.181988
Classical music    0.553627  0.341860  0.451575
Musical            0.262464  0.568232  0.429080
Pop               -0.271719  0.560852 -0.217120
Rock               0.801508 -0.061497 -0.227127
Metal or Hardrock  0.876026 -0.348435 -0.235428
```

```
Punk               0.816995 -0.247060 -0.472734
Hiphop, Rap       -0.374550  0.465859 -0.570731
Reggae, Ska        0.342663  0.340363 -0.349031
Swing, Jazz        0.604564  0.515972  0.191347
Rock n roll        0.780514  0.306515 -0.001735
Alternative        0.737268 -0.003769 -0.025753
Latino             0.043111  0.813650  0.124829
Techno, Trance    -0.234808  0.353415 -0.452851
```

```
                          0         1         2
Dance                           0.665696
Folk                                      0.319220
Country
Classical music                           0.451575
Musical                                   0.429080
Pop                             0.560852
Rock                0.801508
Metal or Hardrock   0.876026
Punk                0.816995
Hiphop, Rap
Reggae, Ska
Swing, Jazz
Rock n roll         0.780514
Alternative         0.737268
Latino                          0.813650
Techno, Trance
```

5.4 Вывод

Данное разбиение дает три легко интерпетируемые группы: в первой любители тяжелой музыик, во вторую попали люди, любящие танцевальную музыку, а третью – любители классики.

6 Классификация через логистическую регрессию

6.1 Постановка задачи

Попробуем ответить на вопрос: что влияет на то, что человек бережливо относится к деньгам?

6.2 Предобработка данных

```
In [1]: import pandas as pd
        import numpy as np

        import matplotlib.pyplot as plt
        import seaborn as sns
        %matplotlib inline
        plt.style.use('fivethirtyeight')
```

Загрузим данные

```
In [3]:
        df1 = pd.read_csv('responses.csv')
```

Выделим подмножество значений, которые будем использовать в дальнейшем исследовании

```
In [5]:mov_mus   = df1.iloc[:,[0,19]]
        scared    = df1.iloc[:,63:73]
        interests = df1.iloc[:,31:63]
        demo      = df1.iloc[:,140:150]
        spending  = df1.iloc[:,134:140]
        predict   = df1.iloc[:,133]

        scared.fillna(0, inplace=True)
        scared = scared.mean(axis=1)

        df2 = mov_mus.join([scared, interests, demo, spending, predict])
        df2.rename(columns={0:'Scared'}, inplace=True)
```

```
In [6]: df2.info()
```

Получили такое подмножество

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1010 entries, 0 to 1009
Data columns (total 52 columns):
Music                          1007 non-null float64
Movies                         1004 non-null float64
Scared                         1010 non-null float64
History                        1008 non-null float64
Psychology                     1005 non-null float64
Politics                       1009 non-null float64
Mathematics                    1007 non-null float64
Physics                        1007 non-null float64
Internet                       1006 non-null float64
PC                             1004 non-null float64
Economy Management             1005 non-null float64
Biology                        1004 non-null float64
Chemistry                      1000 non-null float64
Reading                        1004 non-null float64
Geography                      1001 non-null float64
Foreign languages              1005 non-null float64
Medicine                       1005 non-null float64
Law                            1009 non-null float64
Cars                           1006 non-null float64
Art exhibitions                1004 non-null float64
Religion                       1007 non-null float64
Countryside, outdoors          1003 non-null float64
```

```
Dancing                      1007 non-null float64
Musical instruments          1009 non-null float64
Writing                      1004 non-null float64
Passive sport                995 non-null float64
Active sport                 1006 non-null float64
Gardening                    1003 non-null float64
Celebrities                  1008 non-null float64
Shopping                     1008 non-null float64
Science and technology       1004 non-null float64
Theatre                      1002 non-null float64
Fun with friends             1006 non-null float64
Adrenaline sports            1007 non-null float64
Pets                         1006 non-null float64
Age                          1003 non-null float64
Height                       990 non-null float64
Weight                       990 non-null float64
Number of siblings           1004 non-null float64
Gender                       1004 non-null object
Left - right handed          1007 non-null object
Education                    1009 non-null object
Only child                   1008 non-null object
Village - town               1006 non-null object
House - block of flats       1006 non-null object
Shopping centres             1008 non-null float64
Branded clothing             1008 non-null float64
Entertainment spending       1007 non-null float64
Spending on looks            1007 non-null float64
Spending on gadgets          1010 non-null int64
Spending on healthy eating   1008 non-null float64
Finances                     1007 non-null float64
dtypes: float64(45), int64(1), object(6)
memory usage: 410.4+ KB
```

Обработаем пропущенные данные

```
In [7]:
        drop_list = ['Gender','Left - right handed','Education','Only child','Village - town','H
        df2.dropna(subset= drop_list, inplace=True)
        df2.fillna(0, inplace=True)
```

и представим наглядно те данные с которыми будем работать

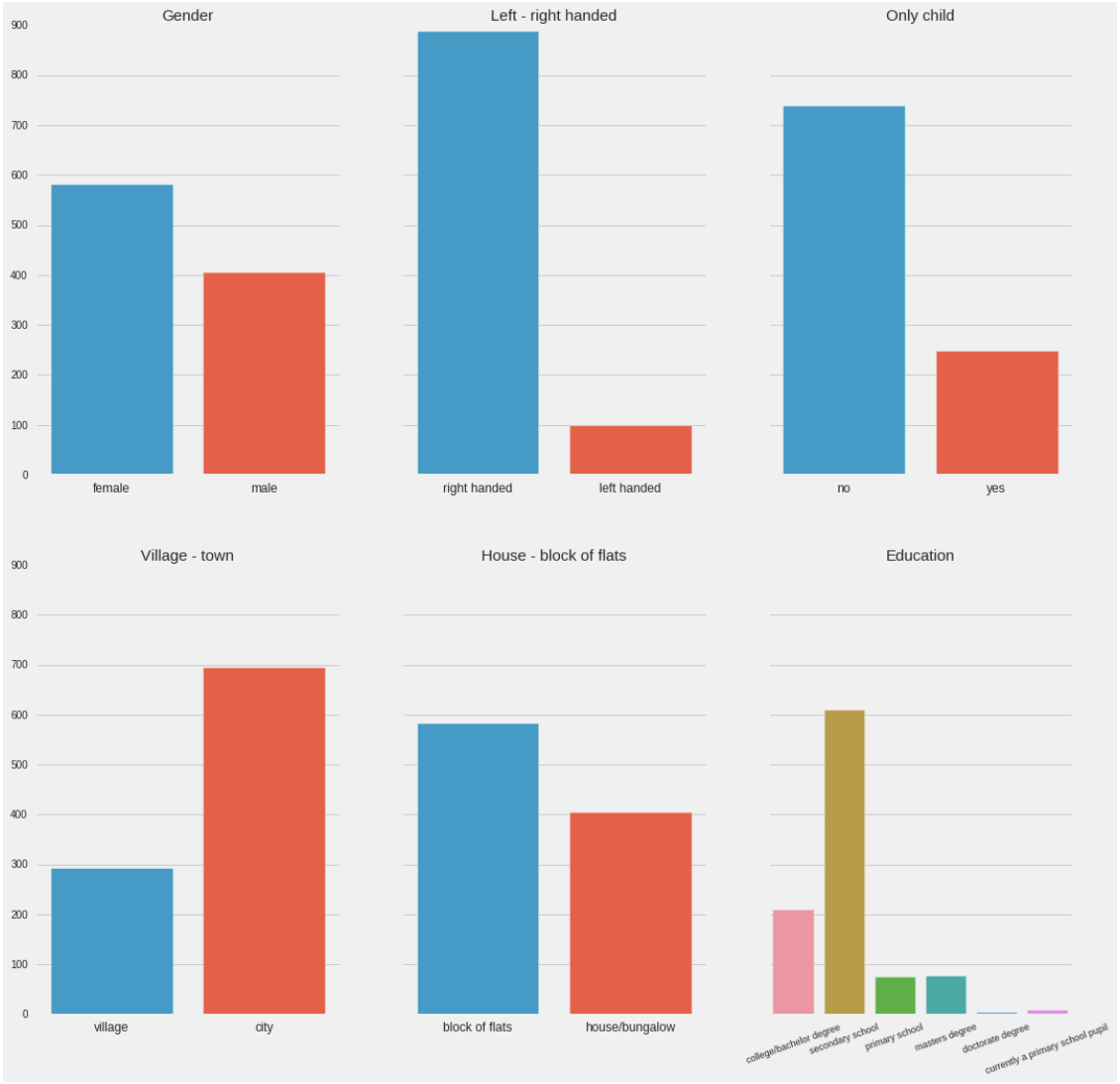
```
In [10]: plt_dict = {}

         for i in range(0,len(interests.columns)):
             plt_dict.update({i:interests.columns[i]})

         fig, ax = plt.subplots(4,8,figsize=(16,16), sharey=True, sharex=True)
```

```
initial = 0

for i in range(4):
    for j in range(8):
        sns.countplot(df2[plt_dict[initial]], ax=ax[i,j])
        ax[i,j].set_ylabel('')
        ax[i,j].set_xlabel('')
        ax[i,j].set_xticklabels(labels=np.arange(0,6), fontsize=5)
        ax[i,j].set_yticklabels(labels=np.arange(0,601,100), fontsize=5)
        ax[i,j].set_title(plt_dict[initial], fontsize=10)
        ax[i,j].set_xlim(.5,5.5)
        ax[i,j].set_ylim(0,600)
        initial += 1
```

Часть данных представленна номинальными переменными. Выведем их

In [13]:
```
obj_dict = {0:'Gender', 1:'Left - right handed', 2:'Only child', 3:'Village - town', 4:
fig, ax = plt.subplots(2,3, figsize=(16,16), sharey=True)

initial = 0

for i in range(2):
    for j in range(3):
        sns.countplot(df2[obj_dict[initial]], ax=ax[i,j])
        ax[i,j].set_title(obj_dict[initial], fontsize=15)
        ax[i,j].set_xlabel('')
        ax[i,j].set_ylabel('')
        ax[i,j].set_xticklabels(labels=df2[obj_dict[initial]].unique(), fontsize=12)
        initial += 1

ax[1,2].set_xticklabels(labels=df2['Education'].unique(), rotation=20, fontsize=9)
```

Out[13]:
```
[<matplotlib.text.Text at 0x7f3d38ecfd30>,
 <matplotlib.text.Text at 0x7f3d38aa6588>,
 <matplotlib.text.Text at 0x7f3d3917bcc0>,
 <matplotlib.text.Text at 0x7f3d3880bf60>,
 <matplotlib.text.Text at 0x7f3d3938f550>,
 <matplotlib.text.Text at 0x7f3d38ae7ef0>]
```



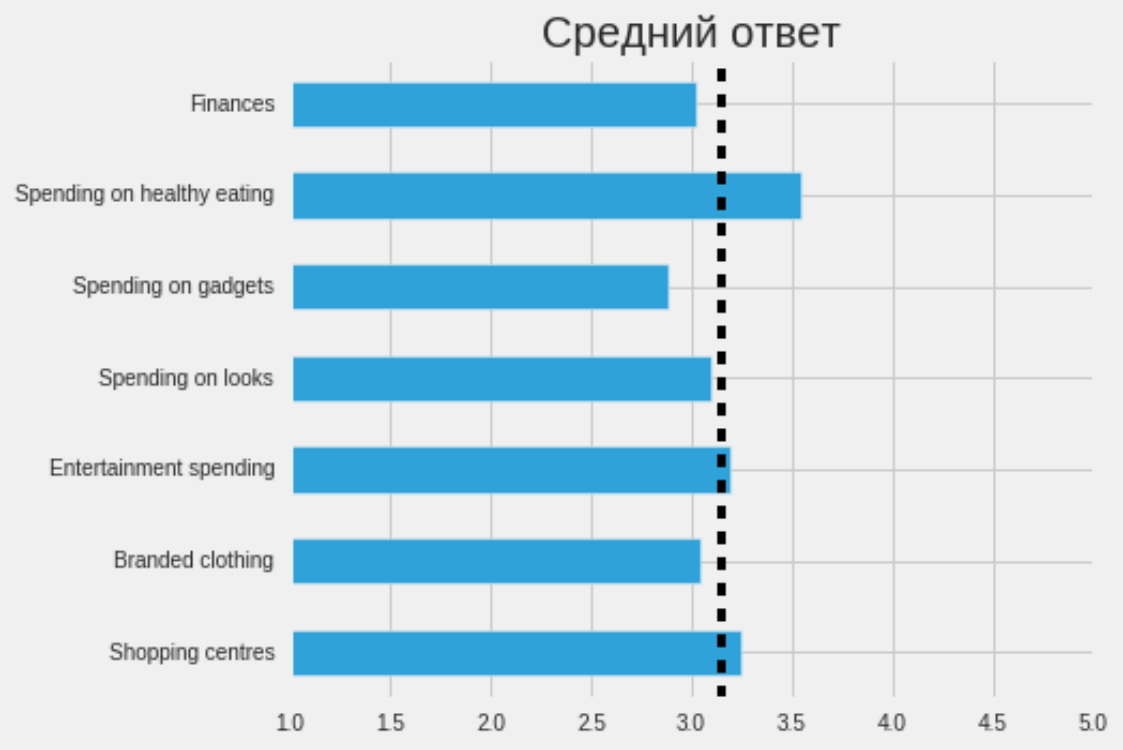
и преобразуем к целочисленным переменным, чтобы их можно было обрабатывать

```
In [14]: gender  = pd.get_dummies(df2['Gender'])
         handed  = pd.get_dummies(df2['Left - right handed'])
         child   = pd.get_dummies(df2['Only child'])
         vil_tow = pd.get_dummies(df2['Village - town'])
         resid   = pd.get_dummies(df2['House - block of flats'])
         educa   = pd.get_dummies(df2['Education'])

         df2.drop(['Gender','Left - right handed','Only child','Village - town','House - block o
         df2 = df2.join([gender, handed, child, vil_tow, resid, educa])
```

```
In [27]: #отобразим характер расходовы по категориям
         spend_aves = pd.Series(df2[spending.columns].mean())
         spend_aves = spend_aves.append(pd.Series(df2['Finances'].mean(), index=['Finances']))
```

```
spend_aves.plot(
    figsize=(6,5), kind='barh', title='Средний ответ',
    color=["#30a2da","#fc4f30","#e5ae38","#6d904f","#8b8b8b",'m', 'r'], xlim=(1,5))

plt.axvline(x=np.mean(spend_aves), color='k', lw=4, ls='dashed')

print('Студенты оценили свою способность сберегать в {:.2f} - но данные по всей таблице
      .format(df2['Finances'].mean(),np.mean(spend_aves)))
```

```
Студенты оценили свою способность сберегать в 3.03 - но данные по всей таблице выше 3.15.
 Это говорит о том, что молодежь тратит денег больше, чем думает, что тратит
```



Чтобы не проводить многозначное прогнозирование, разобьем характеристику "финансы"на 2 группы – 3 и меньше, и 4 и больше. Таким образом, мы сводим задачу к задаче построения логистической регрессии.

```
In [28]:  df2.loc[df2['Finances'] <= 3, 'Finances'] = 0
          df2.loc[df2['Finances'] > 3, 'Finances'] = 1
```

6.3 Идея метода

Логистическая регрессия применяется для предсказания вероятности возникновения некоторого события по значениям множества признаков. Для этого вводится так называемая зависимая

переменная y, принимающая лишь одно из двух значений — как правило, это числа 4 (событие не произошло) и 1 (событие произошло), и множество независимых переменных (также называемых признаками, предикторами или регрессорами) — вещественных $x_1, x_2, \ldots, x_n$, на основе значений которых требуется вычислить вероятность принятия того или иного значения зависимой переменной.

Делается предположение о том, что вероятность наступления события $y = 1$ равна:

$$\mathbb{P}\{y = 1 \mid x\} = f(z),,$$

где $z = \theta^T x = \theta_1 x_1 + \ldots + \theta_n x_n$, x и θ — векторы-столбцы значений независимых переменных $x_1, ldots, x_n$ и параметров (коэффициентов регрессии) — вещественных чисел $\theta_1, ..., \theta_n$, соответственно, а $f(z)$ — так называемая логистическая функция (иногда также называемая сигмоидом или логит-функцией):

$$f(z) = \frac{1}{1 + e^{-z}}.$$

Для подбора параметров $\theta_1, ..., \theta_n$ необходимо составить обучающую выборку, состоящую из наборов значений независимых переменных и соответствующих им значений зависимой переменной y. Формально, это множество пар $(x^{(1)}, y^{(1)}), ..., (x^{(m)}, y^{(m)})$, где $x^{(i)} \in \mathbb{R}^n$ — вектор значений независимых переменных, а $y^{(i)} \in \{0, 1\}$ — соответствующее им значение y. Каждая такая пара называется обучающим примером.

Воспользуемся окружением scikit learn kFold позволяет разделить данные 2 две пары выборок – одну обучающую и вторую контрольную. Для этого подготовим данные, перенеся столбец "финансы"в другую переменную. Класс GridSearchCV позволяет определить оптимальный параметр для логистической регрессии. Затем обучим нашу модель и сравним средную точность на тренировочной выборке и на тестовой. Полученная точность не является идеальной, так как является "платой"за простоту модели. Затем отобразим полученные важности характеристик, и, для удобства, выведем первые 10.

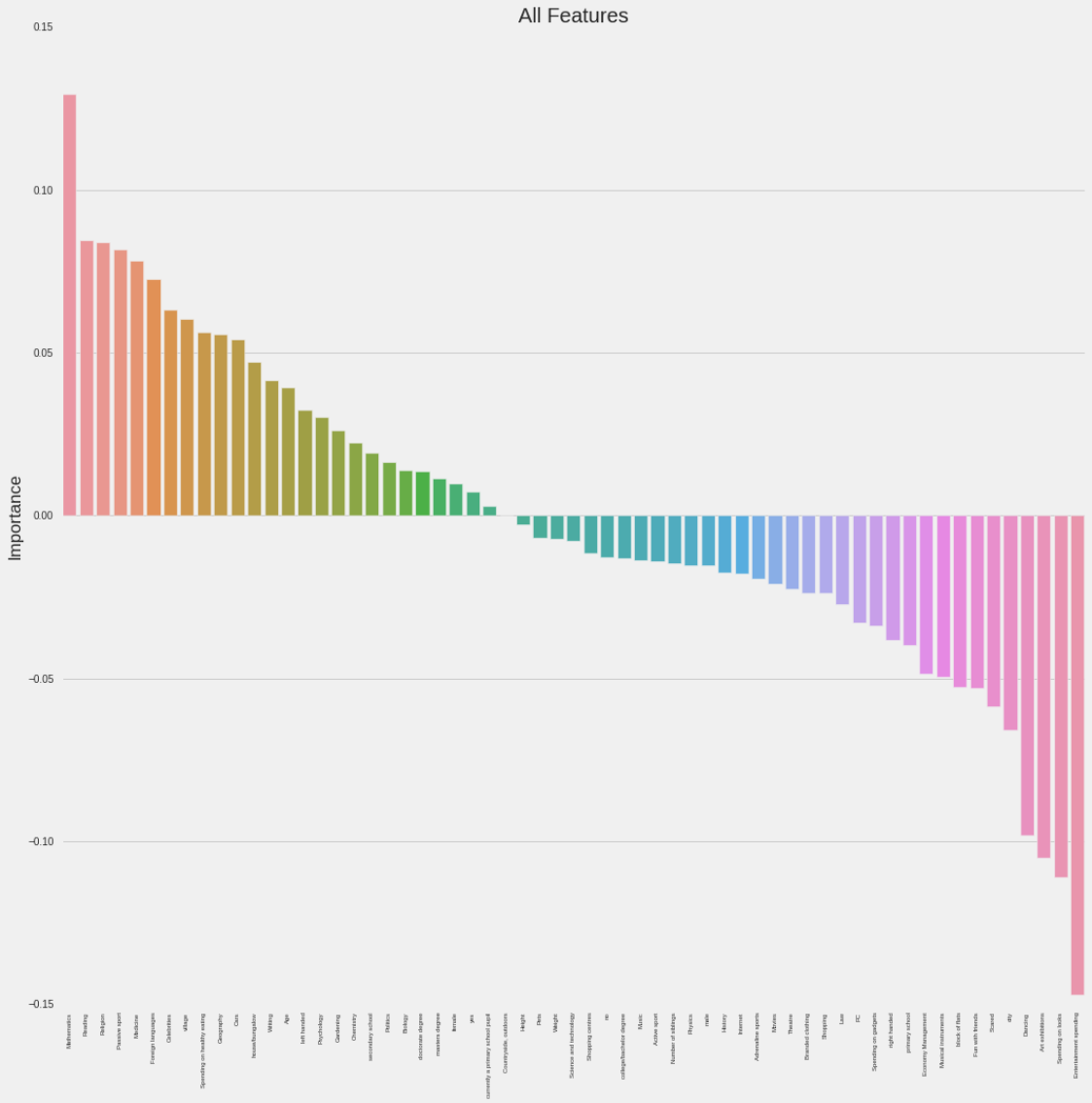
6.4 Реализация

```
In [29]: #ML
         from sklearn.cross_validation import KFold, train_test_split, cross_val_score
         from sklearn.model_selection import GridSearchCV
         from sklearn.linear_model import LogisticRegression
```

```
/usr/local/lib/python3.4/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This
  "This module will be removed in 0.20.", DeprecationWarning)
```

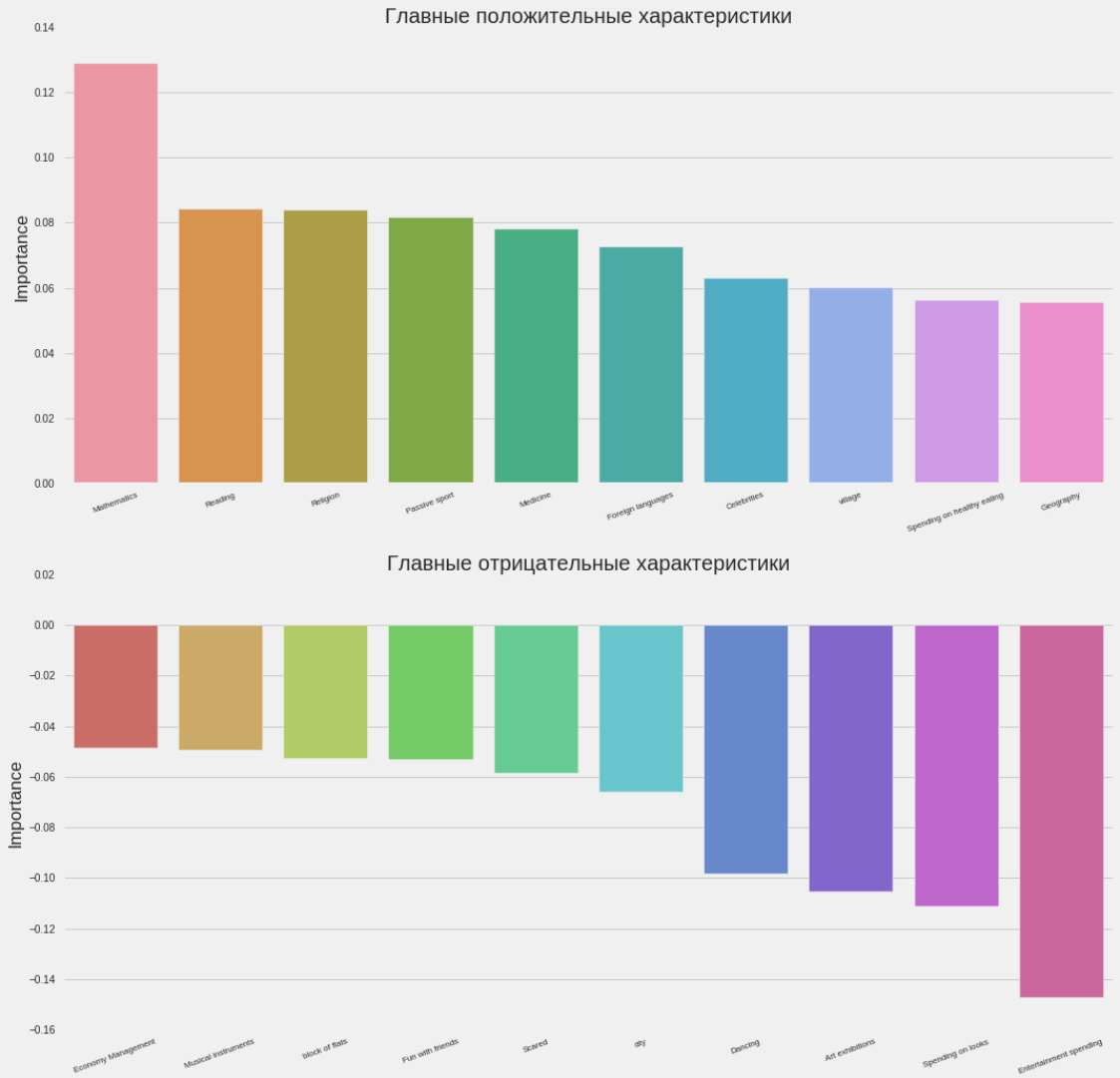
```
In [30]: #set up data for modeling
         x = df2.drop('Finances', axis=1)
         y = df2['Finances']

         x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.3)

         kf = KFold(len(x_train), n_folds=5)
```

```
In [31]: #Use GridSearchCV for parameter tuning
         logreg = LogisticRegression()

         param_grid = {'C':[.01,.03,.1,.3,1,3,10]}

         gs_logreg = GridSearchCV(logreg, param_grid=param_grid, cv=kf)

         gs_logreg.fit(x_train, y_train)
         gs_logreg.best_params_

Out[31]: {'C': 0.01}

In [32]: #fit Logistic Regression model, eval scoring
         logreg = LogisticRegression(C=.01)
         logreg.fit(x_train, y_train)

         print('Average accuracy score on cv (KFold) set: {:.3f}'.format(np.mean(cross_val_score
         print('Accuracy score on test set is: {:.3f}'.format(logreg.score(x_test, y_test)))

Average accuracy score on cv (KFold) set: 0.638
Accuracy score on test set is: 0.670


In [34]: #plot feature importance
         coeff_df = pd.DataFrame(data=logreg.coef_[0], index=[x_train.columns], columns=['Featur
         coeff_df = coeff_df.sort_values(by='Feature_Import', ascending=False)

         fig, ax1 = plt.subplots(1,1, figsize=(16,16))

         sns.barplot(x=coeff_df.index, y=coeff_df['Feature_Import'], ax=ax1)
         ax1.set_title('All Features')
         ax1.set_xticklabels(labels=coeff_df.index, size=6, rotation=90)
         ax1.set_ylabel('Importance')

Out[34]: <matplotlib.text.Text at 0x7f3d38df20f0>
```



In [36]:
```
fig, (ax1, ax2) = plt.subplots(2,1, figsize=(16,16))

sns.barplot(x=coeff_df.index[:10], y=coeff_df['Feature_Import'].head(10), ax=ax1)
ax1.set_title('Главные положительные характеристики')
ax1.set_ylabel('Importance')
ax1.set_xticklabels(labels=coeff_df.index[:10], fontsize=8, rotation=20)


sns.barplot(x=coeff_df.index[-10:], y=coeff_df['Feature_Import'].tail(10), ax=ax2, pale
ax2.set_title('Главные отрицательные характеристики')
ax2.set_ylabel('Importance')
ax2.set_xticklabels(labels=coeff_df.index[-10:], fontsize=8, rotation=20)
```

```
Out[36]: [<matplotlib.text.Text at 0x7f3d38bdd438>,
          <matplotlib.text.Text at 0x7f3d386f63c8>,
          <matplotlib.text.Text at 0x7f3d3873e9e8>,
          <matplotlib.text.Text at 0x7f3d38731518>,
          <matplotlib.text.Text at 0x7f3d3870d048>,
          <matplotlib.text.Text at 0x7f3d3870db38>,
          <matplotlib.text.Text at 0x7f3d38726668>,
          <matplotlib.text.Text at 0x7f3d38729198>,
          <matplotlib.text.Text at 0x7f3d38729c88>,
          <matplotlib.text.Text at 0x7f3d3871e7b8>]
```



6.5 Вывод

Из представленных данных можно сделать некоторые выводы.

Городская молодежь обычно тратит больше денег, чем деревенские жители, что легко объясняется большим количеством мест, где можно потратить деньги.

Так же можно выделить самые затраные привычки (например, танцы или предпочтение тратить деньги на внешний вид). Нельзя не отметить, что пристрастие к чтению, любовь к языкам, географии, математике, химии и другие "интеллектуальные"занятия присущи бережливым людям.

7 Анализ по группам

Проанализируем, как отличаются ответы девушек и мужчин. Для этого выведем средние по двум группам, а также среднее для всей выборки.

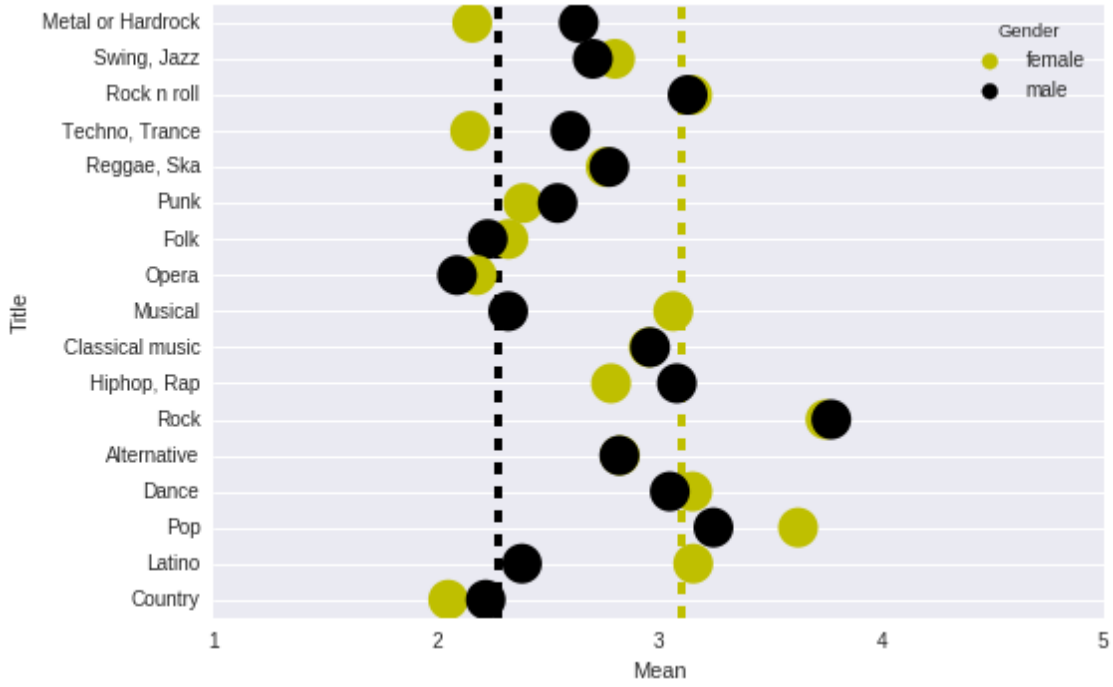
Как видно, девушки больше любят танцевальную музыку, в то время как мальчики предпочитают тяжелую музыку.

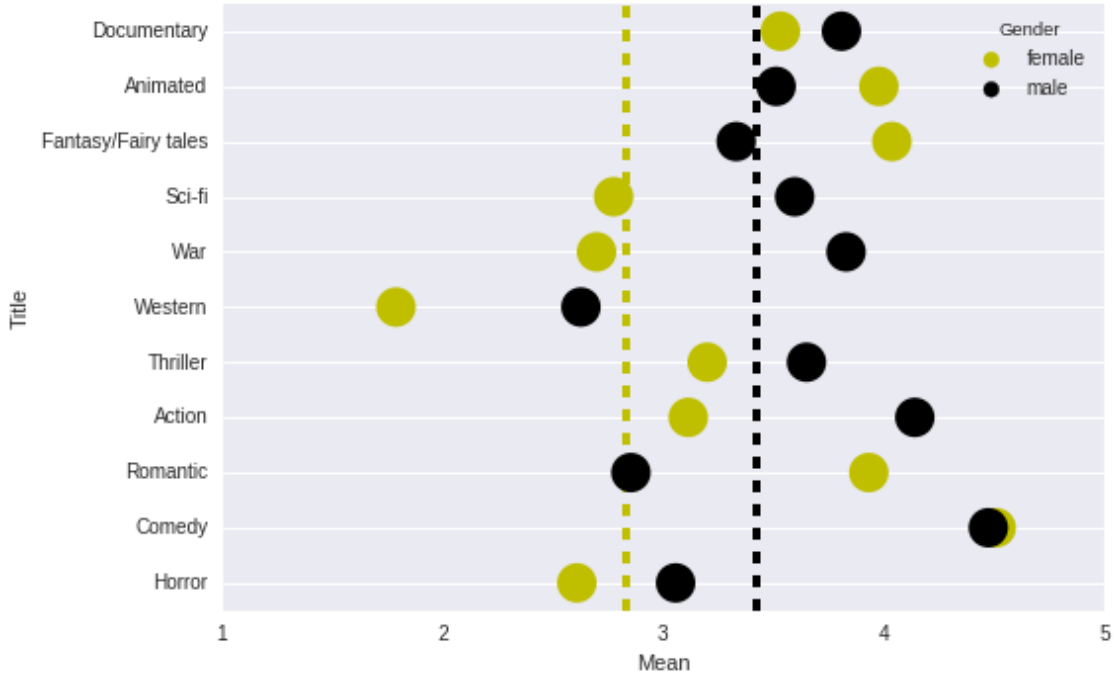
В общем, девушки предпочитают романтические фильмы. а мужчинам нравится экшенкино.

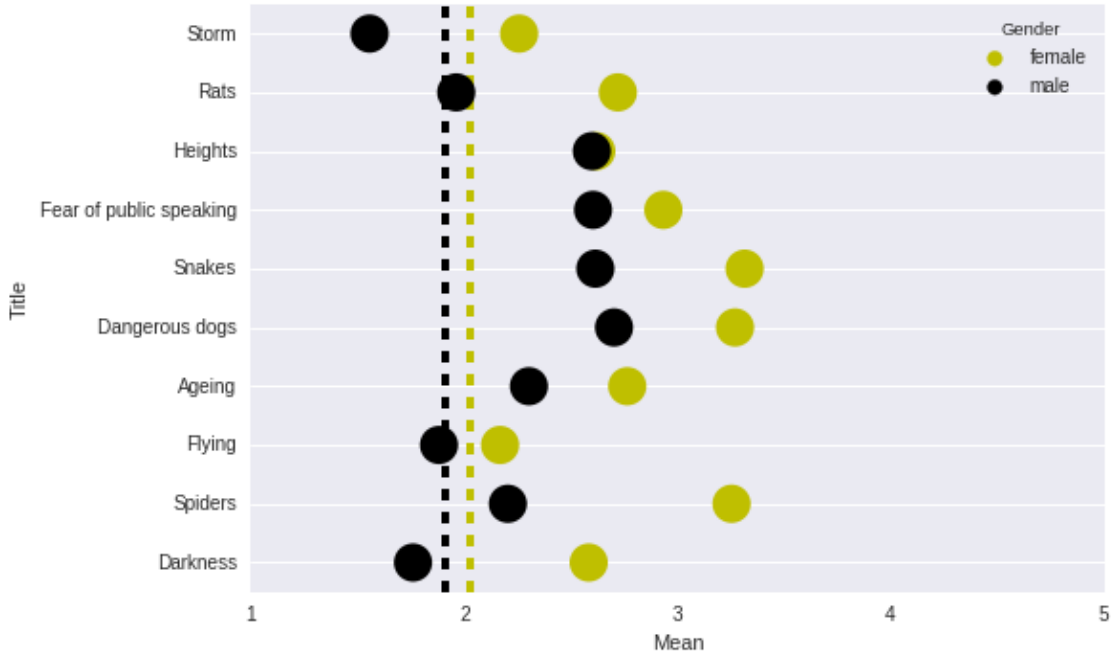
```
In [1]: import pandas as pd
        from pandas import Series
        def customDescribe(x):
            data1 = [x.mean(), x.std(), x.min(), x.quantile(0.25), x.median(),
                     x.quantile(0.75), x.max(), x.skew(), x.kurtosis(), x.mode().max(), x.isnull(
            names = ['mean', 'std', 'min', '25%', '50%', '75%', 'max', 'skewness', 'kurtosis', '
            return Series(data, index=names)



        df = pd.read_csv('responses.csv')
        x = df.loc[:, 'Dance':'Opera'].join(df['Gender']).groupby('Gender')
        x = x.mean()
        x.iloc[1]

Out[1]: Dance                3.048900
        Folk                 2.230392
        Country              2.221130
        Classical music      2.960688
        Musical              2.321951
        Pop                  3.245098
        Rock                 3.775061
        Metal or Hardrock    2.639024
        Punk                 2.544335
        Hiphop, Rap          3.080488
        Reggae, Ska          2.776961
        Swing, Jazz          2.702439
        Rock n roll          3.128954
        Alternative          2.821951
        Latino               2.384236
        Techno, Trance       2.601966
```

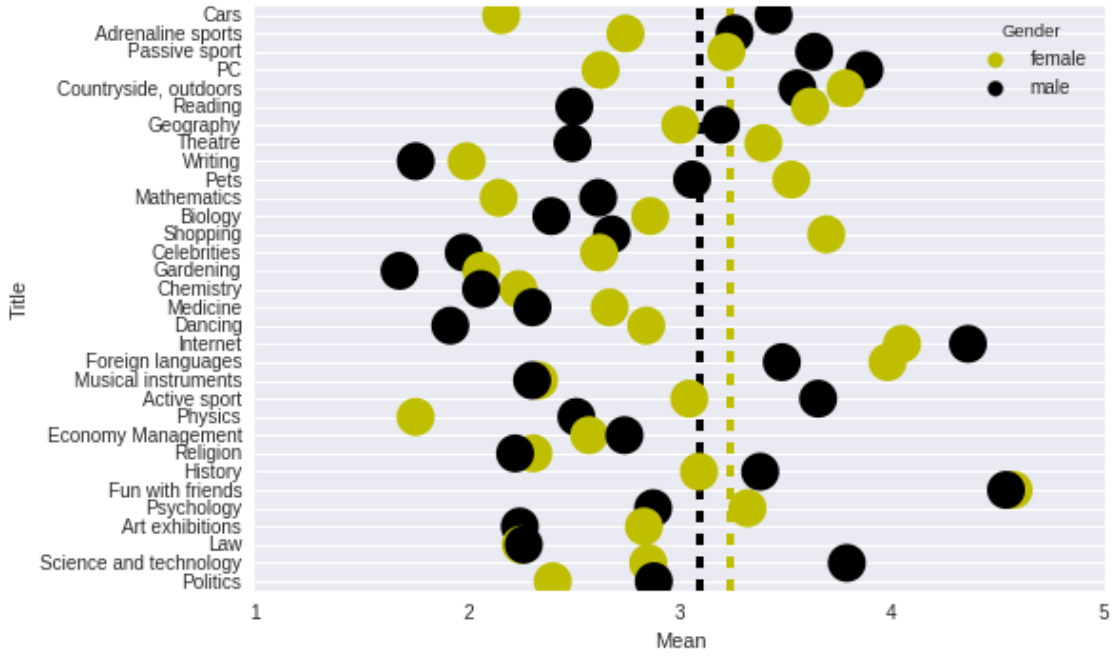
```
Opera                2.092457
Name: male, dtype: float64
```

In [19]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

def analyze_group_differences (df, group, begin, end, plt_title):
    groups = df.loc[:,begin:end].join(df[group]).groupby(group).mean()
    x= [i for i in  groups.iterrows()]
    res = []
    for i,s in x:
        for k,v in s.to_dict().items():
            res.append((i, k, v))
    new_df = pd.DataFrame(data = res, columns = [ 'Gender', 'Title', 'Mean'])
    fig = plt.figure()
    ax = fig.gca()
    sns.stripplot(data=new_df, x = 'Mean', y = 'Title', hue= 'Gender', orient='h', pale

                  size = 20, ax = ax);
    plt.axvline(x=np.mean(groups.mean().iloc[0]), color='y', lw=4, ls='dashed')
    plt.axvline(x=np.mean(groups.mean().iloc[1]), color='k', lw=4, ls='dashed')
    ax.set_xticks([1,2,3,4,5])


    plt.grid()
    plt.show()

analyze_group_differences(df=df, group ='Gender', begin = 'Dance', end = 'Opera', plt_t
analyze_group_differences(df=df, group ='Gender', begin = 'Horror', end = 'Action', plt
analyze_group_differences(df=df, group ='Gender', begin = 'Flying', end = 'Fear of publ
analyze_group_differences(df=df, group ='Gender', begin = 'History', end = 'Pets', plt_
```


Metal or Hardrock
Swing, Jazz
Rock n roll
Techno, Trance
Reggae, Ska
Punk
Folk
Opera
Musical
Classical music
Hiphop, Rap
Rock
Alternative
Dance
Pop
Latino
Country
Title
1
2
3
4
5
Mean
Gender
female
male


Documentary
Animated
Fantasy/Fairy tales
Sci-fi
War
Western
Thriller
Action
Romantic
Comedy
Horror
Title
1
2
3
4
5
Mean
Gender
female
male


Storm
Rats
Heights
Fear of public speaking
Snakes
Dangerous dogs
Ageing
Flying
Spiders
Darkness
Title
Gender
female
male
1
2
3
4
5
Mean


Cars
Adrenaline sports
Passive sport
PC
Countryside, outdoors
Reading
Geography
Theatre
Writing
Pets
Mathematics
Biology
Shopping
Celebrities
Gardening
Chemistry
Medicine
Dancing
Internet
Foreign languages
Musical instruments
Active sport
Physics
Economy Management
Religion
History
Fun with friends
Psychology
Art exhibitions
Law
Science and technology
Politics
Title
Gender
female
male
1
2
3
4
5
Mean

In [ ]: