

**Московский Государственный Технический Университет имени Н.Э. Баумана**

**Факультет «Информатика и системы управления»**

**Кафедра ИУ-5**

**«Системы обработки информации и управления»**

**Отчет по лабораторной работе № 1**

**Разведочный анализ данных.  
Исследование и визуализация данных.**

**по дисциплине**

**«Методы машинного обучения»**

Выполнил студент группы ИУ5-22М:  
Бурашников В.В.

**Москва, 2020 г.**

## 1) Текстовое описание набора данных

В качестве набора данных мы будем использовать набор данных по обнаружению присутствия людей в помещении - <https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+> (<https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+>)

Эта задача является очень актуальной для создания "умных зданий", которые выполняют все требования по кондиционированию воздуха, температурным условиям, но при этом экономят электроэнергию в том случае, если людей в помещении нет.

Датасет состоит из трех файлов:

- all\_seasons.csv - обучающая выборка (в этом примере используется только данный файл)

Каждый файл содержит следующие колонки:

- age - Age of the player
- pts - Average number of points scored
- ast - Average number of assists distributed
- net\_rating - Team's point differential per 100 possessions while the player is on the court
- ast\_pct - Percentage of teammate field goals the player assisted while he was on the floor

## Импорт библиотек

Импортируем библиотеки с помощью команды import. Как правило, все команды import размещают в первой ячейке ноутбука, но мы в этом примере будем подключать все библиотеки последовательно, по мере их использования.

In [65]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

## Загрузка данных

Загрузим файлы датасета в помощью библиотеки Pandas.

Файл имеет формат CSV (<https://ru.wikipedia.org/wiki/CSV> (<https://ru.wikipedia.org/wiki/CSV>)). Часто в файлах такого формата в качестве разделителей используются символы ",", ";" или табуляция. Поэтому вызывая метод read\_csv всегда стоит явно указывать разделитель данных с помощью параметра sep. Чтобы узнать какой разделитель используется в файле его рекомендуется предварительно посмотреть в любом текстовом редакторе.

In [66]:

```
# Будем анализировать данные только на обучающей выборке
data = pd.read_csv('/home/vladimir/all_seasons.csv', sep=",")
```

## 2) Основные характеристики датасета

In [67]:

```
# Первые 5 строк датасета
data.head()
```

Out[67]:

	Unnamed: 0	age	pts	ast	net_rating	ast_pct
0	0	36	5.7	3.1	16.1	0.113
1	1	28	2.3	0.3	12.3	0.048
2	2	39	0.8	0.4	-2.1	0.148
3	3	24	3.7	0.6	-8.7	0.077
4	4	34	2.4	0.2	-11.2	0.040

In [68]:

```
# Размер датасета - 8143 строк, 7 колонок
data.shape
```

Out[68]:

(11145, 6)

In [69]:

```
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
```

Всего строк: 11145

In [70]:

```
# Список колонок
data.columns
```

Out[70]:

Index(['Unnamed: 0', 'age', 'pts', 'ast', 'net\_rating', 'ast\_pct'], dtype='object')

In [71]:

```
# Список колонок с типами данных
data.dtypes
```

Out[71]:

Unnamed: 0 int64  
age int64  
pts float64  
ast float64  
net\_rating float64  
ast\_pct float64  
dtype: object

In [72]:

```
# Проверим наличие пустых значений
# Цикл по колонкам датасета
for col in data.columns:
    # Количество пустых значений - все значения заполнены
    temp_null_count = data[data[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

Unnamed: 0 - 0  
age - 0  
pts - 0  
ast - 0  
net\_rating - 0  
ast\_pct - 0

In [73]:

```
# Основные статистические характеристики набора данных
data.describe()
```

Out[73]:

	Unnamed: 0	age	pts	ast	net_rating	ast_pct
count	11145.000000	11145.000000	11145.000000	11145.000000	11145.000000	11145.000000
mean	5572.000000	27.168686	8.126487	1.801463	-2.153899	0.131078
std	3217.428709	4.344164	5.935482	1.789940	12.150611	0.095017
min	0.000000	18.000000	0.000000	0.000000	-200.000000	0.000000
25%	2786.000000	24.000000	3.500000	0.600000	-6.300000	0.065000
50%	5572.000000	27.000000	6.600000	1.200000	-1.300000	0.102000
75%	8358.000000	30.000000	11.500000	2.400000	3.200000	0.178000
max	11144.000000	44.000000	36.100000	11.700000	300.000000	1.000000

In [74]:

```
# Определим уникальные значения для целевого признака
data['age'].unique()
```

Out[74]:

```
array([36, 28, 39, 24, 34, 38, 25, 29, 27, 31, 26, 21, 23, 32, 30, 22, 35,
       37, 33, 18, 20, 40, 43, 19, 41, 42, 44])
```

Целевой признак является бинарным и содержит только значения 0 и 1.

### 3) Визуальное исследование датасета

Для визуального исследования могут быть использованы различные виды диаграмм, мы построим только некоторые варианты диаграмм, которые используются достаточно часто.

#### Диаграмма рассеяния ([https://en.wikipedia.org/wiki/Scatter\\_plot](https://en.wikipedia.org/wiki/Scatter_plot))

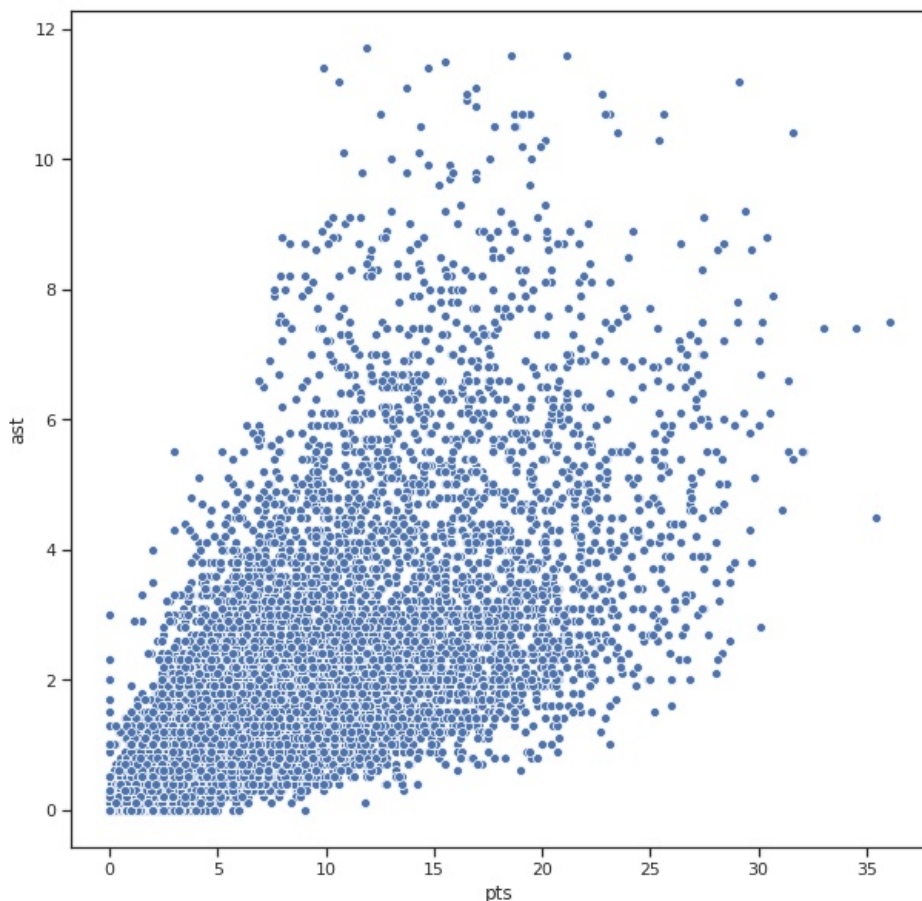
Позволяет построить распределение двух колонок данных и визуально обнаружить наличие зависимости. Не предполагается, что значения упорядочены (например, по времени).

In [75]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='pts', y='ast', data=data)
```

Out[75]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855fc5a7f0>



Можно видеть что между полями pts и ast присутствует почти линейная зависимость.

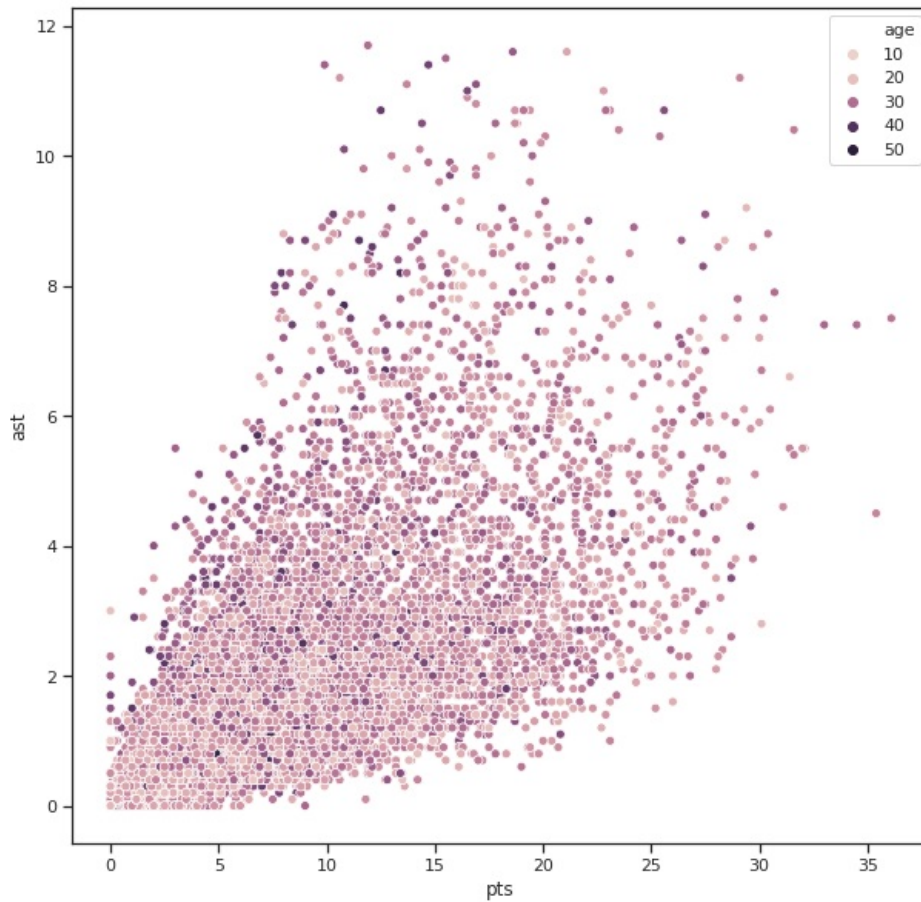
Посмотрим насколько на эту зависимость влияет целевой признак.

In [76]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='pts', y='ast', data=data, hue='age')
```

Out[76]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f85670b0160>



## [Гистограмма \(https://en.wikipedia.org/wiki/Histogram\)](https://en.wikipedia.org/wiki/Histogram)

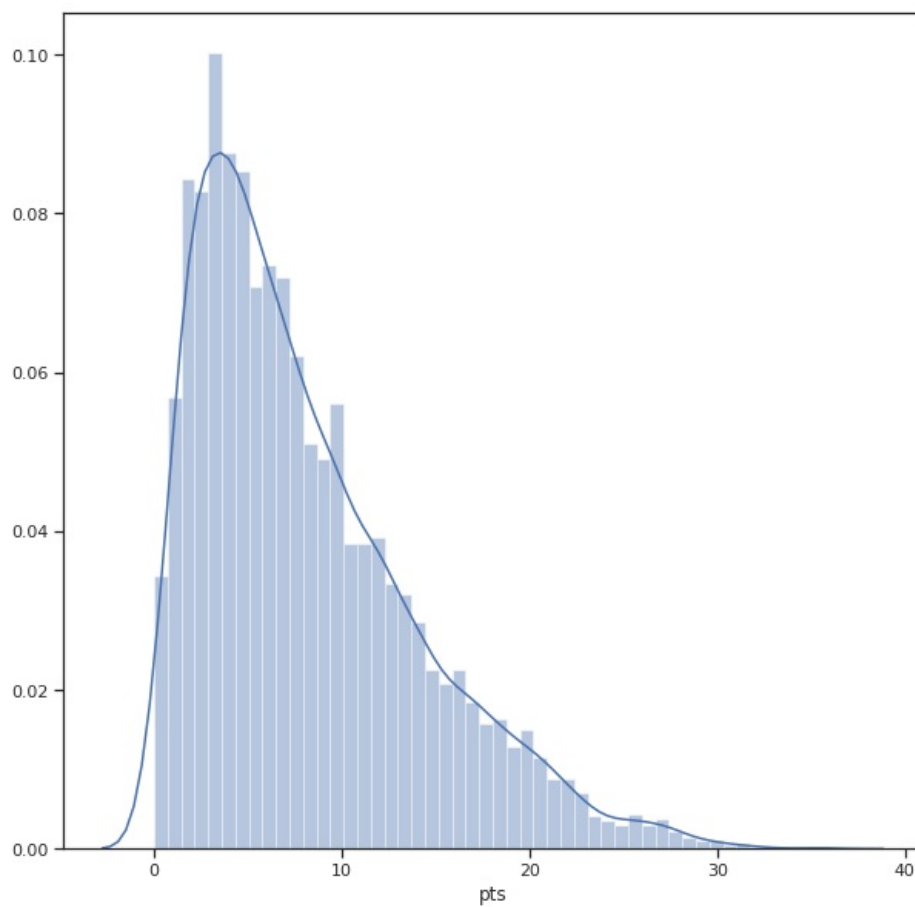
Позволяет оценить плотность вероятности распределения данных.

In [77]:

```
fig, ax = plt.subplots(figsize=(10,10))
sns.distplot(data['pts'])
```

Out[77]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855fc864a8>



## Jointplot

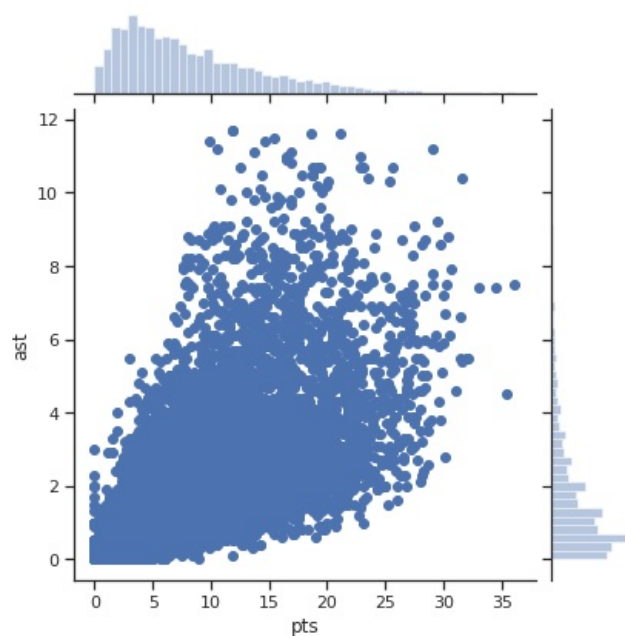
Комбинация гистограмм и диаграмм рассеивания.

In [78]:

```
sns.jointplot(x='pts', y='ast', data=data)
```

Out[78]:

<seaborn.axisgrid.JointGrid at 0x7f855fc73358>

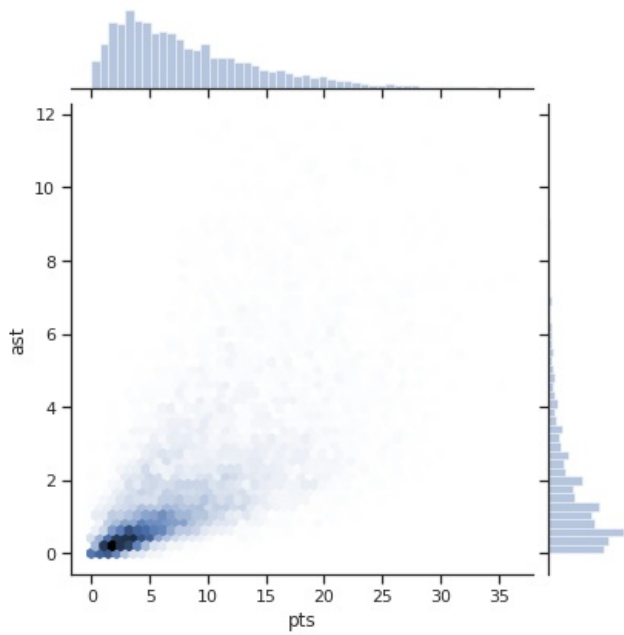


In [79]:

```
sns.jointplot(x='pts', y='ast', data=data, kind="hex")
```

Out[79]:

<seaborn.axisgrid.JointGrid at 0x7f855fb0ca90>

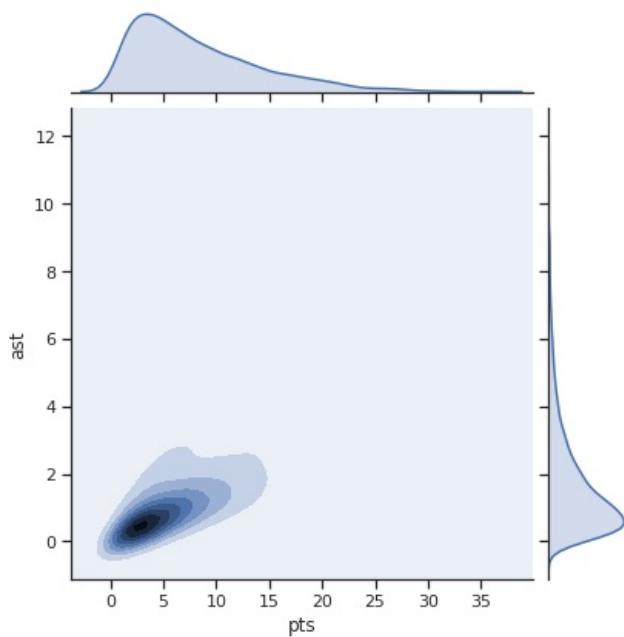


In [80]:

```
sns.jointplot(x='pts', y='ast', data=data, kind="kde")
```

Out[80]:

<seaborn.axisgrid.JointGrid at 0x7f855f90b550>



## "Парные диаграммы"

Комбинация гистограмм и диаграмм рассеивания для всего набора данных.

Выводится матрица графиков. На пересечении строки и столбца, которые соответствуют двум показателям, строится диаграмма рассеивания. В главной диагонали матрицы строятся гистограммы распределения соответствующих показателей.

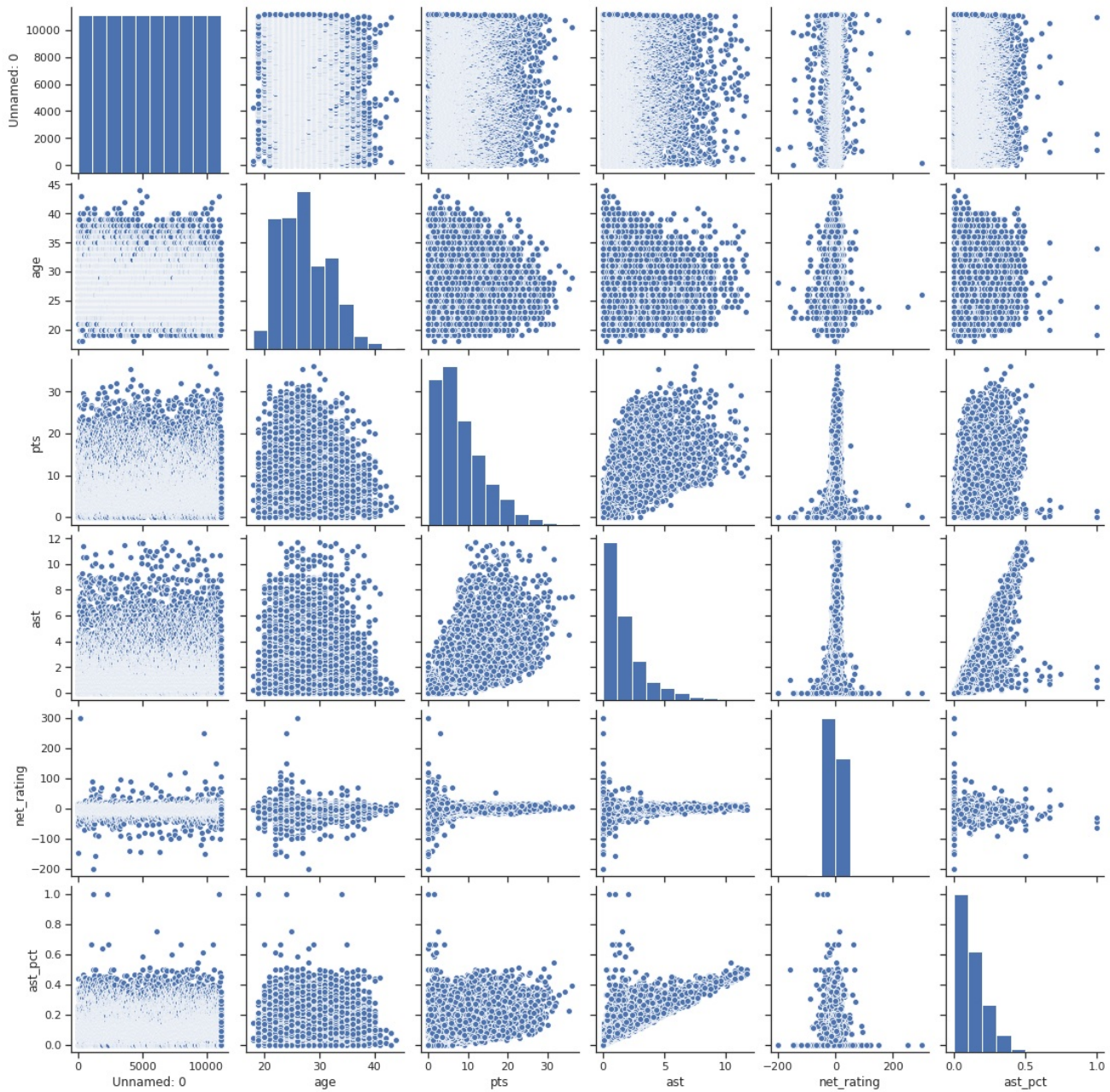


In [81]:

```
sns.pairplot(data)
```

Out[81]:

<seaborn.axisgrid.PairGrid at 0x7f855f7af3c8>



С помощью параметра "hue" возможна группировка по значениям какого-либо признака.

In [82]:

```
sns.pairplot(data, hue="age")
```



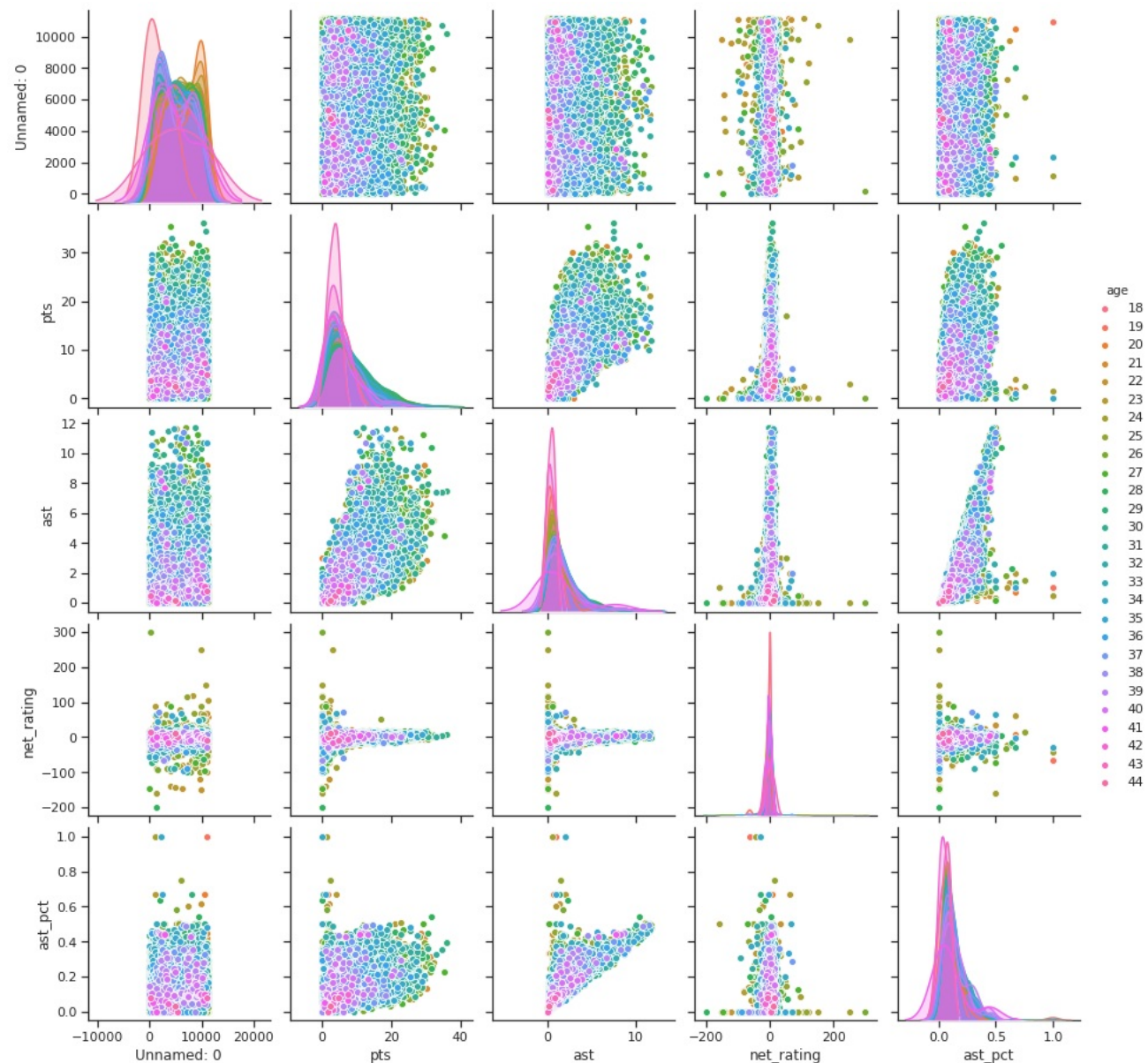
```

/home/vladimir/PycharmProjects/giis_lab1/env/lib/python3.6/site-packages/seaborn/distributions.py:28
8: UserWarning: Data must have variance to compute a kernel density estimate.
warnings.warn(msg, UserWarning)
/home/vladimir/PycharmProjects/giis_lab1/env/lib/python3.6/site-packages/seaborn/distributions.py:28
8: UserWarning: Data must have variance to compute a kernel density estimate.
warnings.warn(msg, UserWarning)
/home/vladimir/PycharmProjects/giis_lab1/env/lib/python3.6/site-packages/seaborn/distributions.py:28
8: UserWarning: Data must have variance to compute a kernel density estimate.
warnings.warn(msg, UserWarning)
/home/vladimir/PycharmProjects/giis_lab1/env/lib/python3.6/site-packages/seaborn/distributions.py:28
8: UserWarning: Data must have variance to compute a kernel density estimate.
warnings.warn(msg, UserWarning)
/home/vladimir/PycharmProjects/giis_lab1/env/lib/python3.6/site-packages/seaborn/distributions.py:28
8: UserWarning: Data must have variance to compute a kernel density estimate.
warnings.warn(msg, UserWarning)

```

Out[82]:

<seaborn.axisgrid.PairGrid at 0x7f855ea695c0>



[Ящик с усами \(https://en.wikipedia.org/wiki/Box\\_plot\)](https://en.wikipedia.org/wiki/Box_plot)

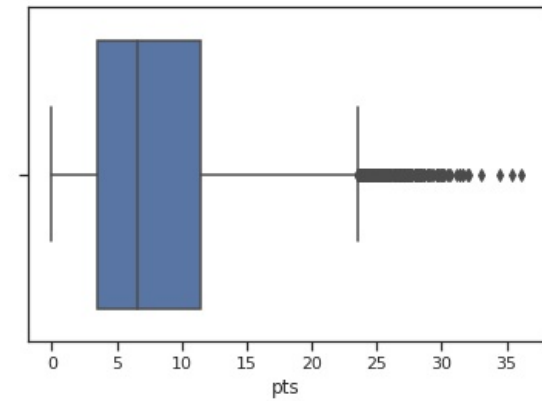
Отображает одномерное распределение вероятности.

In [83]:

```
sns.boxplot(x=data['pts'])
```

Out[83]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855e50d518>

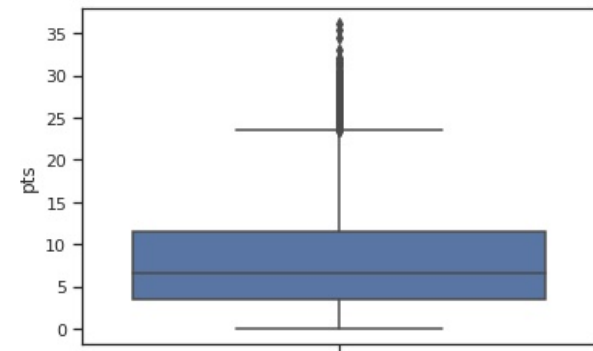


In [84]:

```
# По вертикали  
sns.boxplot(y=data['pts'])
```

Out[84]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855d3df400>

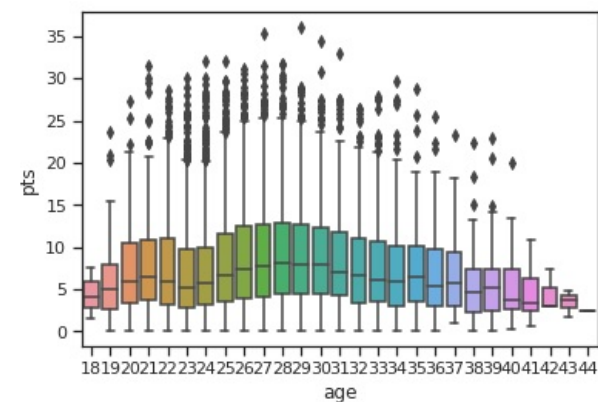


In [85]:

```
# Распределение параметра Humidity сгруппированные по Occurance.  
sns.boxplot(x='age', y='pts', data=data)
```

Out[85]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855d38af28>



## Violin plot ([https://en.wikipedia.org/wiki/Violin\\_plot](https://en.wikipedia.org/wiki/Violin_plot))

Похоже на предыдущую диаграмму, но по краям отображаются распределения плотности -

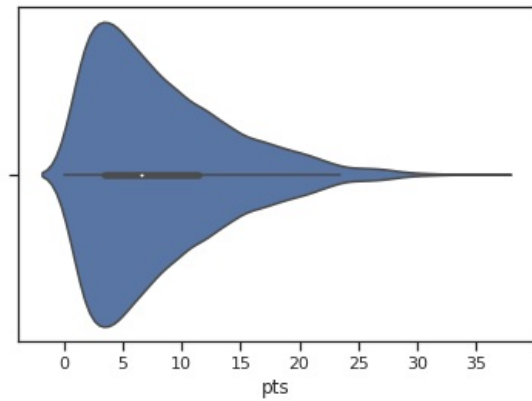
[https://en.wikipedia.org/wiki/Kernel\\_density\\_estimation](https://en.wikipedia.org/wiki/Kernel_density_estimation) ([https://en.wikipedia.org/wiki/Kernel\\_density\\_estimation](https://en.wikipedia.org/wiki/Kernel_density_estimation))

In [86]:

```
sns.violinplot(x=data['pts'])
```

Out[86]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855cb38f60>

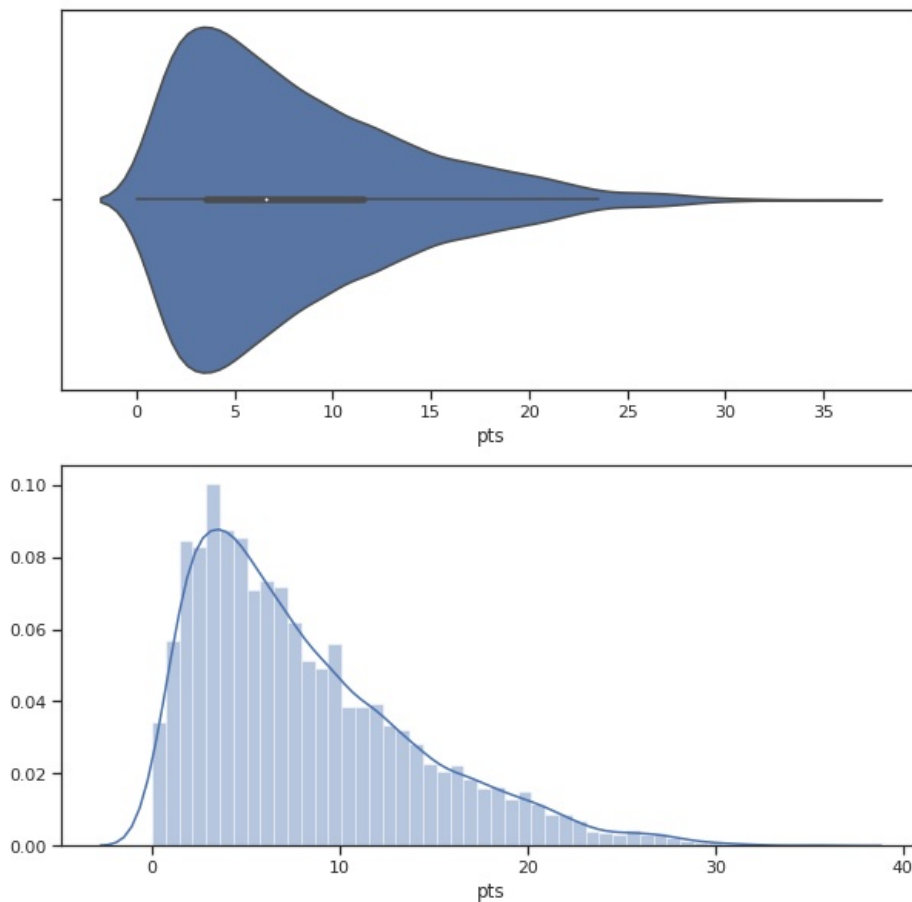


In [87]:

```
fig, ax = plt.subplots(2, 1, figsize=(10,10))  
sns.violinplot(ax=ax[0], x=data['pts'])  
sns.distplot(data['pts'], ax=ax[1])
```

Out[87]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855caf0d30>



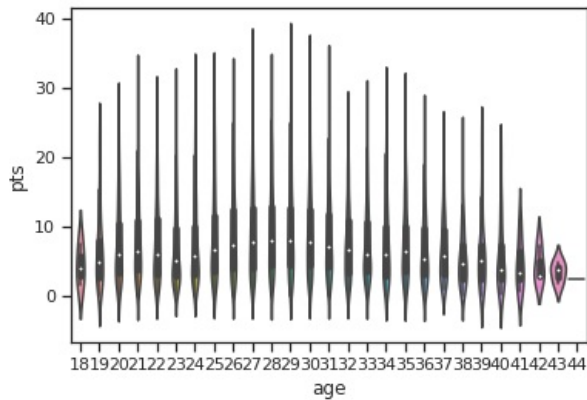
Из приведенных графиков видно, что violinplot действительно показывает распределение плотности.

In [88]:

```
# Распределение параметра pts сгруппированные по age.  
sns.violinplot(x='age', y='pts', data=data)
```

Out[88]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855cae9748>

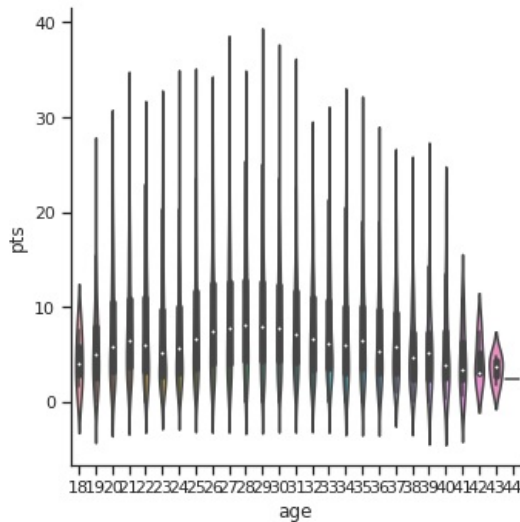


In [89]:

```
sns.catplot(y='pts', x='age', data=data, kind="violin", split=True)
```

Out[89]:

<seaborn.axisgrid.FacetGrid at 0x7f855cce0d68>



## 4) Информация о корреляции признаков

Проверка корреляции признаков позволяет решить две задачи:

1. Понять какие признаки (колонки датасета) наиболее сильно коррелируют с целевым признаком (в нашем примере это колонка "Оссуирасу"). Именно эти признаки будут наиболее информативными для моделей машинного обучения. Признаки, которые слабо коррелируют с целевым признаком, можно попробовать исключить из построения модели, иногда это повышает качество модели. Нужно отметить, что некоторые алгоритмы машинного обучения автоматически определяют ценность того или иного признака для построения модели.
2. Понять какие нецелевые признаки линейно зависимы между собой. Линейно зависимые признаки, как правило, очень плохо влияют на качество моделей. Поэтому если несколько признаков линейно зависимы, то для построения модели из них выбирают какой-то один признак.

In [90]:

```
data.corr()
```

Out[90]:

	Unnamed: 0	age	pts	ast	net_rating	ast_pct
Unnamed: 0	1.000000	-0.123543	0.034429	0.013879	0.011620	-0.007876
age	-0.123543	1.000000	-0.000810	0.083695	0.088161	0.063037
pts	0.034429	-0.000810	1.000000	0.652098	0.217883	0.322214
ast	0.013879	0.083695	0.652098	1.000000	0.165236	0.802008
net_rating	0.011620	0.088161	0.217883	0.165236	1.000000	0.060158
ast_pct	-0.007876	0.063037	0.322214	0.802008	0.060158	1.000000

Корреляционная матрица содержит коэффициенты корреляции между всеми парами признаков.

Корреляционная матрица симметрична относительно главной диагонали. На главной диагонали расположены единицы (корреляция признака самого с собой).

На основе корреляционной матрицы можно сделать следующие выводы:

- Целевой признак наиболее сильно коррелирует с ast (0.08) и net\_rating (0.088). Эти признаки обязательно следует оставить в модели.
- Целевой признак отчасти коррелирует с ast\_pct (0.63). Этот признак стоит также оставить в модели.

Описание метода corr - <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html> (<https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html>)

По умолчанию при построении матрицы используется коэффициент корреляции [Пирсона](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D1%8F) (<https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D1%8F>). Возможно также построить корреляционную матрицу на основе коэффициентов корреляции Кендалла и Спирмена. На практике три метода редко дают значимые различия.

In [91]:

```
data.corr(method='pearson')
```

Out[91]:

	Unnamed: 0	age	pts	ast	net_rating	ast_pct
Unnamed: 0	1.000000	-0.123543	0.034429	0.013879	0.011620	-0.007876
age	-0.123543	1.000000	-0.000810	0.083695	0.088161	0.063037
pts	0.034429	-0.000810	1.000000	0.652098	0.217883	0.322214
ast	0.013879	0.083695	0.652098	1.000000	0.165236	0.802008
net_rating	0.011620	0.088161	0.217883	0.165236	1.000000	0.060158
ast_pct	-0.007876	0.063037	0.322214	0.802008	0.060158	1.000000

In [92]:

```
data.corr(method='kendall')
```

Out[92]:

	Unnamed: 0	age	pts	ast	net_rating	ast_pct
Unnamed: 0	1.000000	-0.085828	0.028374	0.022138	0.005544	0.003288
age	-0.085828	1.000000	0.027252	0.084977	0.102135	0.051564
pts	0.028374	0.027252	1.000000	0.556109	0.199133	0.253036
ast	0.022138	0.084977	0.556109	1.000000	0.167148	0.663608
net_rating	0.005544	0.102135	0.199133	0.167148	1.000000	0.065114
ast_pct	0.003288	0.051564	0.253036	0.663608	0.065114	1.000000

In [93]:

```
data.corr(method='spearman')
```

Out[93]:

	Unnamed: 0	age	pts	ast	net_rating	ast_pct
Unnamed: 0	1.000000	-0.124735	0.042534	0.032288	0.008406	0.004363
age	-0.124735	1.000000	0.040898	0.122228	0.147235	0.074851
pts	0.042534	0.040898	1.000000	0.744875	0.283870	0.360461
ast	0.032288	0.122228	0.744875	1.000000	0.238981	0.827434
net_rating	0.008406	0.147235	0.283870	0.238981	1.000000	0.095005
ast_pct	0.004363	0.074851	0.360461	0.827434	0.095005	1.000000

В случае большого количества признаков анализ числовой корреляционной матрицы становится неудобен.

Для визуализации корреляционной матрицы будем использовать "тепловую карту" heatmap которая показывает степень корреляции различными цветами.

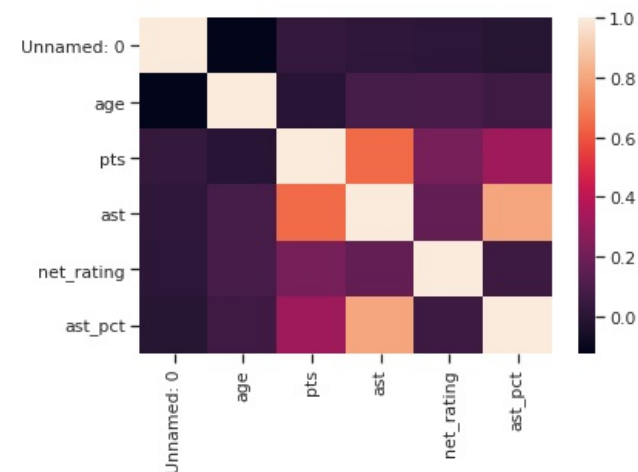
Используем метод heatmap библиотеки seaborn - <https://seaborn.pydata.org/generated/seaborn.heatmap.html>  
(<https://seaborn.pydata.org/generated/seaborn.heatmap.html>)

In [94]:

```
sns.heatmap(data.corr())
```

Out[94]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855c7e39e8>

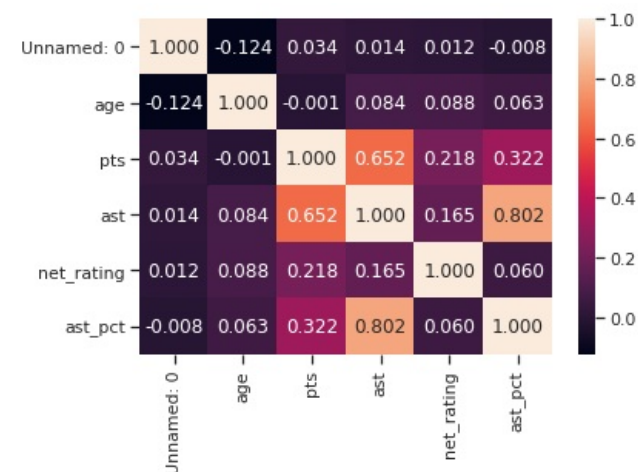


In [95]:

```
# Вывод значений в ячейках  
sns.heatmap(data.corr(), annot=True, fmt='.3f')
```

Out[95]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855c5bdb00>

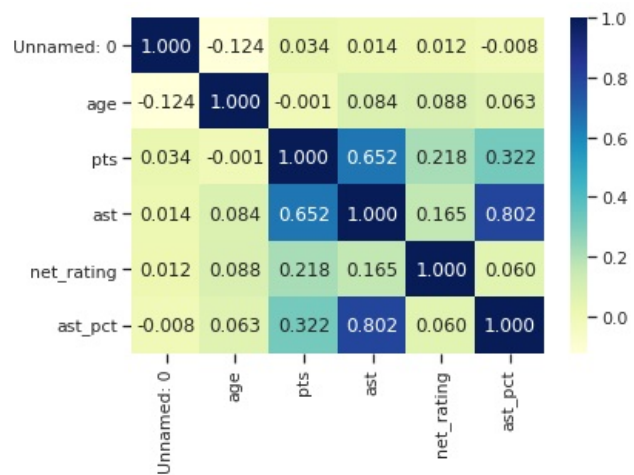


In [96]:

```
# Изменение цветовой гаммы
sns.heatmap(data.corr(), cmap='YlGnBu', annot=True, fmt='.3f')
```

Out[96]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855c50cba8>

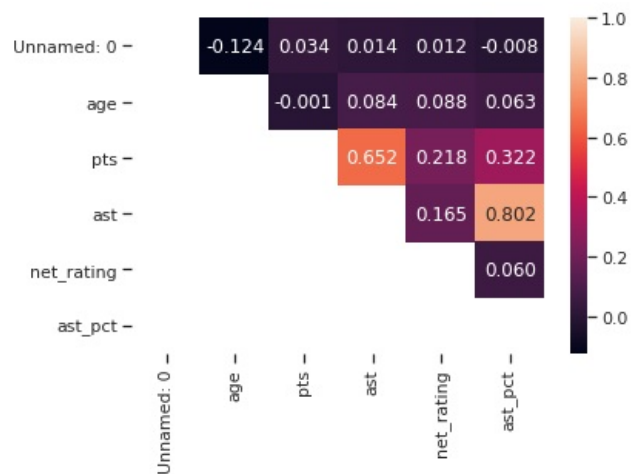


In [100]:

```
# Треугольный вариант матрицы
mask = np.zeros_like(data.corr(), dtype=np.bool)
# чтобы оставить нижнюю часть матрицы
# mask[np.triu_indices_from(mask)] = True
# чтобы оставить верхнюю часть матрицы
mask[np.tril_indices_from(mask)] = True
sns.heatmap(data.corr(), mask=mask, annot=True, fmt='.3f')
```

Out[100]:

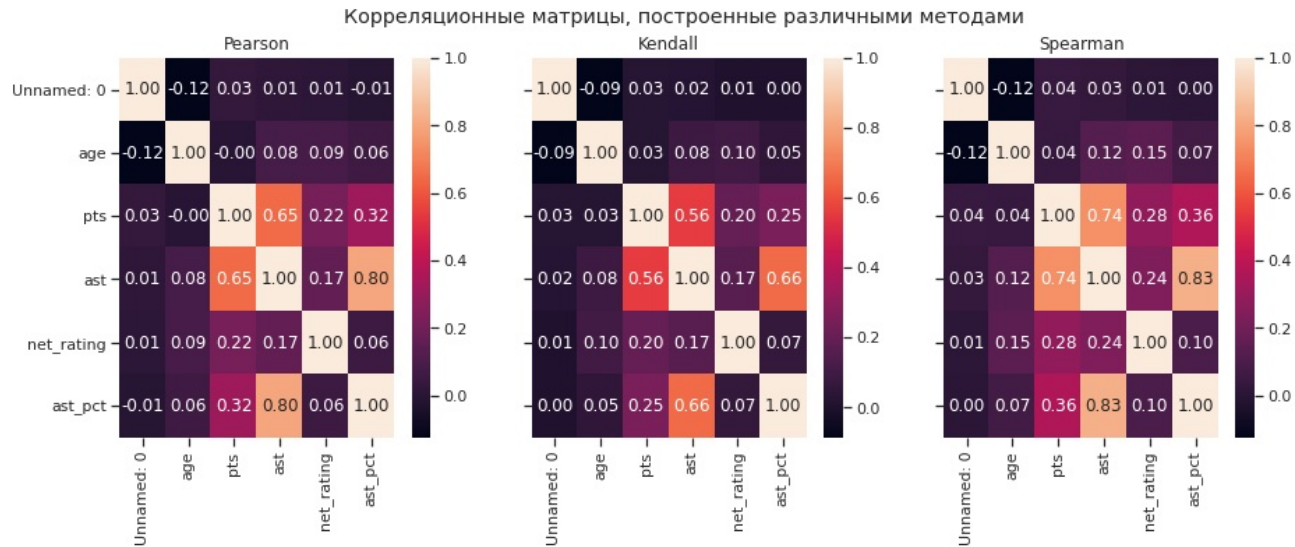
<matplotlib.axes.\_subplots.AxesSubplot at 0x7f855c2faa90>





In [98]:

```
fig, ax = plt.subplots(1, 3, sharex='col', sharey='row', figsize=(15,5))
sns.heatmap(data.corr(method='pearson'), ax=ax[0], annot=True, fmt='.2f')
sns.heatmap(data.corr(method='kendall'), ax=ax[1], annot=True, fmt='.2f')
sns.heatmap(data.corr(method='spearman'), ax=ax[2], annot=True, fmt='.2f')
fig.suptitle('Корреляционные матрицы, построенные различными методами')
ax[0].title.set_text('Pearson')
ax[1].title.set_text('Kendall')
ax[2].title.set_text('Spearman')
```



Необходимо отметить, что тепловая карта не очень хорошо подходит для определения корреляции нецелевых признаков между собой.

В примере тепловая карта помогает определить значимую корреляцию между признаками ast и pts, следовательно только один из этих признаков можно включать в модель.

Но в реальной модели могут быть сотни признаков и коррелирующие признаки могут образовывать группы, состоящие более чем из двух признаков. Увидеть такие группы с помощью тепловой карты сложно.

Для решения задачи предлагается новый вариант визуализации - "Солнечная корреляционная карта" [Solar correlation map](https://github.com/Zapf-Consulting/solar-correlation-map) (<https://github.com/Zapf-Consulting/solar-correlation-map>).

К сожалению, данная библиотека пока работает только через файловый интерфейс и не предназначена для встраивания в ноутбук.

Примеры статей с описанием работы библиотеки:

- <https://www.oreilly.com/learning/a-new-visualization-to-beautifully-explore-correlations> (<https://www.oreilly.com/learning/a-new-visualization-to-beautifully-explore-correlations>)
- <https://www.mtab.com/the-puzzle-of-visualizing-correlations/> (<https://www.mtab.com/the-puzzle-of-visualizing-correlations/>)

## Дополнительные ссылки на обучающие ноутбуки

[The Best Tutorial for Beginners \(Kaggle\)](https://www.kaggle.com/getting-started/71679) (<https://www.kaggle.com/getting-started/71679>)

In [ ]: