**Базовый вариант**:

1. Загрузить данные из CSV-файлов в датафреймы Pandas. Далее по заданию основным датафреймом считается тот, в который загружены данные из orders.csv, с ним идет основная работа.
2. Посмотреть типы данных колонок и количество non-null значений в основном датафрейме с помощью info().
3. Конвертировать колонки «order\_start\_prepare» и «order\_ready» в формат datetime.
4. Посчитать количество пропусков для каждой колонки основного датафрейма. Сделать первые выводы по пунктам выше.
5. Для каждой записи в основном датафрейме необходимо найти соответствующее название филиала на основе его store\_id и добавить это значение в новую колонку «store\_name».  Например, для записи заказа со store\_id  = 5014 значение в колонке «store\_name» будет то, которое указано в stores.csv напротив этого номера.
6. Аналогично создать колонку «product\_name» на основе product\_id.
7. Аналогично создать колонку «region\_name» на основе region\_id.
8. Заменить пропуски в «price» на минимальную цену товара.
9. Создать новую колонку «items\_count», в которой содержится информация о количестве товаров в каждом заказе.
10. Создать колонку «order\_price», равную сумме стоимости всех товаров в заказе. Удалить колонку «price».
11. Построить гистограмму для «product\_name». Сделать вывод.
12. Построить график плотности (kdeplot) для «order\_price». Сделать вывод.
13. Удалить колонку «product\_id», «product\_name» и дубликаты записей. Для каждого заказа должна остаться одна запись.
14. Провести «анализ одной переменной»:

13.1. Посмотреть статистические показатели для колонок с помощью метода describe().

13.2. Для колонок store\_name, profit, delivery\_distance, order\_price, items\_count, planned\_prep\_time, region\_name построить графики: histplot для дискретных признаков, kdeplot (из seaborn) для непрерывных (цена заказа, расстояние, выгода, ожидаемое время).

13.3. Сделать выводы. Колонки с названиями можно удалить.

1. Провести анализ на наличие выбросов в данных.
2. Выбрать стратегию избавления от NaN-значений и аномалий. Обосновать выбор стратегии. Применить стратегию.
3. Создать целевую переменную «prepared\_on\_time» («Заказ выполнен вовремя») — является ли фактическое время заказа (actual\_time) меньше запланированного времени (planned\_prep\_time) + 5 минут? Если да, то заказ считается выполненным вовремя. actual\_time рассчитывается как разница между «order\_ready» и «order\_start\_prepare». При необходимости закодировать в бинарный формат (1 и 0, где 1 - заказ выполнен вовремя). Если создавалась колонка «actual\_time», то удалить.
4. Посчитать корреляцию между всеми числовыми признаками (profit, delivery\_distance, store\_id, planned\_prep\_time, region\_id, items\_count, order\_price) и целевой переменной («prepared\_on\_time»), визуализировать с помощью heatmap.
5. Сделать выводы по каждому пункту выше.

**Продвинутый вариант**:

1. Загрузить данные из CSV-файлов в датафреймы Pandas. Далее по заданию основным датафреймом считается тот, в который загружены данные из orders.csv, с ним идет основная работа.
2. Посмотреть типы данных колонок и количество non-null значений в основном датафрейме с помощью info().
3. Конвертировать колонки «order\_start\_prepare» и «order\_ready» в формат datetime.
4. Посчитать количество пропусков для каждой колонки основного датафрейма. Сделать первые выводы по пунктам выше.
5. Посчитать корреляцию между пропущенными значениями (NaN) по разным столбцам и вывести в формате тепловой карты. Сделать первые выводы по пунктам.
6. Построить графики (гистограммы или графики плотности в зависимости от типа переменной) для всех колонок.
7. Выбрать стратегии избавления от пропусков.
8. Проверить колонки на наличие аномалий и выбрать стратегии избавления от них.
9. Для каждой записи в основном датафрейме необходимо найти соответствующее название филиала на основе его store\_id и добавить это значение в новую колонку «store\_name».  Например, для записи заказа со store\_id  = 5014 значение в колонке «store\_name» будет то, которое указано в stores.csv напротив этого номера.
10. Аналогично создать колонку «product\_name» на основе product\_id.
11. Аналогично создать колонку «region\_name» на основе region\_id.
12. Создать новую колонку «items\_count», в которой содержится информация о количестве продуктов в каждом заказе.
13. Создать колонку «order\_price», равную сумме стоимости всех товаров в заказе. Удалить колонку «price».
14. Построить гистограмму для «product\_name». Сделать вывод.
15. Удалить колонку «product\_id», «product\_name» и дубликаты записей. Для каждого заказа должна остаться одна запись.
16. Получить из «order\_start\_prepare» день недели и внести его в новую колонку «day\_of\_week». При необходимости закодировать.
17. Получить из «order\_start\_prepare» час и минуту, вынести в отдельные колонки «hour\_start» и «minutes\_start».
18. Провести «анализ одной переменной»:

18.1. Посмотреть статистические показатели для колонок с помощью метода describe().

18.2. Для колонок store\_name, delivery\_distance, hour\_start, minutes\_start, order\_price, items\_count, planned\_prep\_time, region\_name, day\_of\_week построить графики: histplot для дискретных признаков, kdeplot (из seaborn) для непрерывных (цена заказа, расстояние, выгода, ожидаемое время).

18.3. Сделать выводы. Колонки с названиями можно удалить.

1. Закодировать «hour\_start» и «minutes\_start» как циклические признаки.
2. Создать целевую переменную «prepared\_on\_time» («Заказ выполнен вовремя») — является ли фактическое время заказа (actual\_time) меньше запланированного времени (planned\_prep\_time) + 5 минут? Если да, то заказ считается выполненным вовремя. actual\_time рассчитывается как разница между «order\_ready» и «order\_start\_prepare». При необходимости закодировать в бинарный формат (1 и 0, где 1 - заказ выполнен вовремя). Если создавалась колонка «actual\_time», то удалить.
3. Посмотреть на баланс классов с помощью value\_counts().
4. Посчитать корреляцию Пирсона между всеми числовыми признаками и целевой переменной. Также посчитать phi\_k корреляцию. Вывести матрицы корреляций в формате heatmap.
5. Сделать выводы по пунктам выше.

Помощь в задании:

1. Получение целевой переменной:

# Вычисление фактического времени выполнения в минутах

df['actual\_time'] = (df['order\_ready'] - df['order\_start\_prepare']).dt.total\_seconds() / 60

# Вычисление таргета на основе фактического и ожидаемого времени

df['prepared\_on\_time'] = df['actual\_time'] <= df['planned\_prep\_time'] + 5

1. Получение для недели:

df['day\_of\_week'] = df['order\_start\_prepare'].dt.dayofweek  # 0: Monday, 6: Sunday