МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Систем обработки информации и управления»

ОТЧЕТ

Рубежный контроль №__1

по дисциплине «Методы машинного обучения»

Тема: «Создание "истории о данных"»

ИСПОЛНИТЕЛЬ: **Ли Яцзинь**

ФИО

группа ИУ5 -21М

подпись

" " 2024 г.

ПРЕПОДАВАТЕЛЬ: Гапанюк Ю .Е

ФИО

_____
подпись

_____

" " 2024 г.

Москва - 2024

Варианты заданий

Номер варианта	Номер задачи №1	Номер задачи №2	дополнительная задача
4 +15=19	19	39	График рассеяния

Задача №19.

Для набора данных проведите масштабирование данных для одного (произвольного) числового признака с использованием метода "Mean Normalisation".

Задача №19

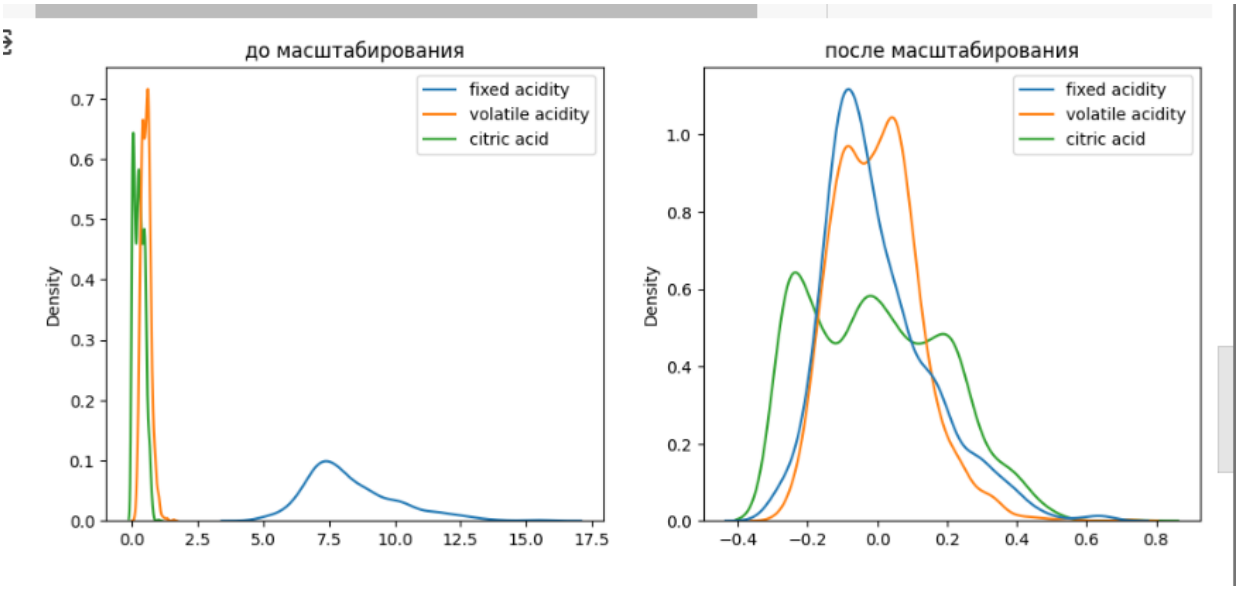
```
# @title Задача №19
#Класс для нормализации среднего значения MeanNormalisation
class MeanNormalisation:

        def fit(self, param_df):
                self.means = X_train.mean(axis=0)
                maxs = X_train.max(axis=0)
                mins = X_train.min(axis=0)
                self.ranges = maxs - mins

        def transform(self, param_df):
                param_df_scaled = (param_df - self.means) / self.ranges
                return param_df_scaled

        def fit_transform(self, param_df):
                self.fit(param_df)
                return self.transform(param_df)
```

```
[19] sc21 = MeanNormalisation()
     data_cs21_scaled = sc21.fit_transform( data)
     data_cs21_scaled.describe()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide
count	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03	1.599000e+03
mean	5.332403e-17	2.166289e-17	-1.777468e-17	-1.110917e-17	1.666376e-17	-7.776421e-18	5.554587e-18
std	1.540793e-01	1.226436e-01	1.948011e-01	9.657042e-02	7.857313e-02	1.473262e-01	1.162379e-01
min	-3.291714e-01	-2.793291e-01	-2.709756e-01	-1.122470e-01	-1.259875e-01	-2.095059e-01	-1.429957e-01
25%	-1.079325e-01	-9.439761e-02	-1.809756e-01	-4.375380e-02	-2.915950e-02	-1.249989e-01	-8.645863e-02
50%	-3.713604e-02	-5.356516e-03	-1.097561e-02	-2.320586e-02	-1.413446e-02	-2.640735e-02	-2.992153e-02
75%	7.790821e-02	7.683527e-02	1.490244e-01	4.191404e-03	4.229480e-03	7.218420e-02	5.488413e-02
max	6.708286e-01	7.206709e-01	7.290244e-01	8.877530e-01	8.740125e-01	7.904941e-01	8.570043e-01

```
# Построение плотности распределения
def draw_kde(col_list, df1, df2, label1, label2):
        fig, (ax1, ax2) = plt.subplots(
                ncols=2, figsize=(12, 5))
        # первый график
        ax1.set_title(label1)
        sns.kdeplot(data=df1[col_list], ax=ax1)
        # второй график
        ax2.set_title(label2)
        sns.kdeplot(data=df2[col_list], ax=ax2)
        plt.show()
```

```
draw_kde(['fixed acidity', 'volatile acidity', 'citric acid'], data, data_cs21_scaled, 'до масштаб
```



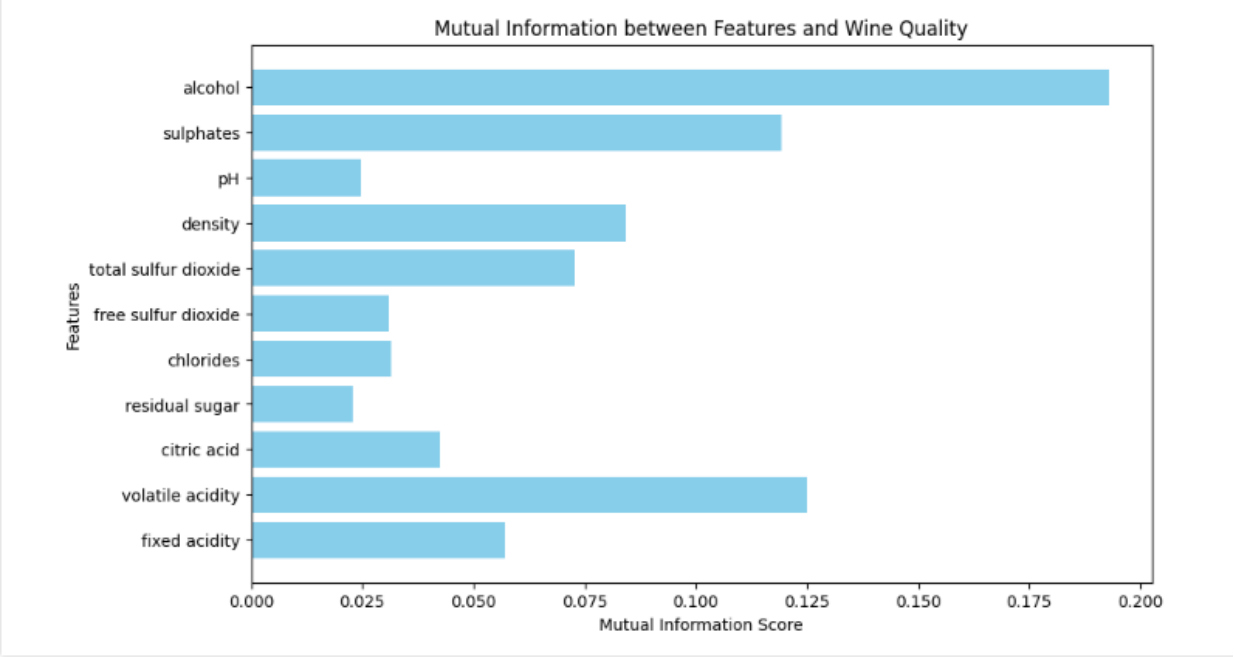
Задача №39

Для набора данных проведите процедуру отбора признаков (feature selection). Используйте класс SelectPercentile для 10% лучших признаков, и метод, основанный на взаимной информации.

```
# @title Задача №39

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.feature_selection import mutual_info_classif


wine_data = pd.read_csv("/content/winequality-red.csv")

print(wine_data.head())
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
```

```
# функции и метки сегментов
X = wine_data.drop('quality', axis=1)   # функции ввода
y = wine_data['quality']   # выходная переменная

# Взаимная информация
mutual_info_scores = mutual_info_classif(X, y)

# Визуализация оценок взаимной информации
plt.figure(figsize=(10, 6))
plt.barh(X.columns, mutual_info_scores, color='skyblue')
plt.xlabel('Mutual Information Score')
plt.ylabel('Features')
plt.title('Mutual Information between Features and Wine Quality')
plt.show()
```




```python
# Выберите объекты с помощью SelectPercentile
selector = SelectPercentile(mutual_info_classif, percentile=10)
X_selected = selector.fit_transform(X, y)

# Получите индекс столбца и имя выбранного объекта.
selected_features_indices = selector.get_support(indices=True)
selected_features_names = X.columns[selected_features_indices]

# Распечатать выбранные объекты
print("Selected features:")
print(X[selected_features_names].head())
```

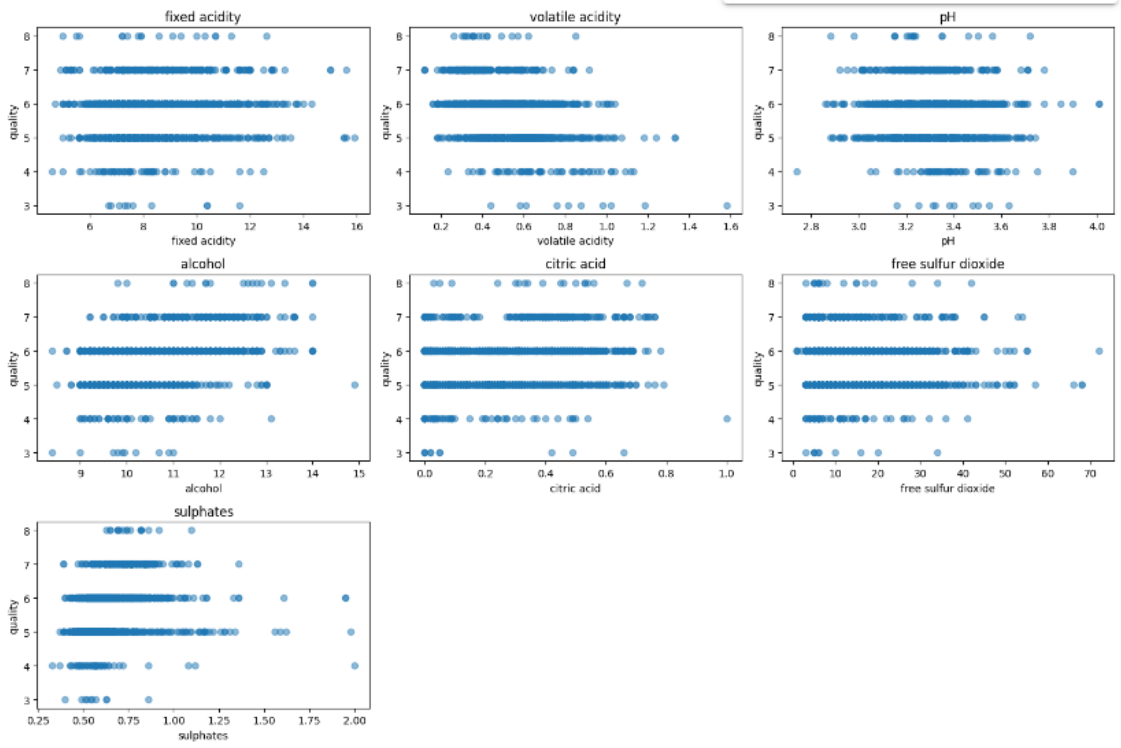
```
Selected features:
   alcohol
0      9.4
1      9.8
2      9.8
3      9.8
4      9.4
```

дополнительная задача

Для студентов групп ИУ5-21М, ИУ5И-21М, ИУ5Ц-21М - для пары произвольных колонок данных построить график "Диаграмма рассеяния".

дополнительная задача

```
# @title дополнительная задача



#Scatter plot
import pandas as pd
import matplotlib.pyplot as plt

# Загрузить набор винных данных
wine_data = pd.read_csv("/content/winequality-red.csv")

# входные переменные
input_variables = ['fixed acidity', 'volatile acidity', 'pH', 'alcohol', 'citric acid', 'free sulfur dioxi

#выходная переменная
output_variable = 'quality'

# Нарисуйте диаграмму рассеяния
plt.figure(figsize=(15, 10))
for i, variable in enumerate(input_variables, 1):
        plt.subplot(3, 3, i)
        plt.scatter(wine_data[variable], wine_data[output_variable], alpha=0.5)
        plt.title(variable)
        plt.xlabel(variable)
        plt.ylabel(output_variable)
plt.tight_layout()
plt.show()
```



Задание

Выбрать набор данных (датасет).

. Создать "историю о данных" в виде юпитер-ноутбука, с учетом следующих требований:

1. История должна содержать не менее 5 шагов (где 5 - рекомендуемое количество шагов). Каждый шаг содержит график и его текстовую интерпретацию.

2. На каждом шаге наряду с удачным итоговым графиком рекомендуется в юпитер-ноутбуке оставлять результаты предварительных

"неудачных" графиков.

3. Не рекомендуется повторять виды графиков, желательно создать 5 графиков различных видов.

4. Выбор графиков должен быть обоснован использованием методологии data-to-viz. Рекомендуется учитывать типичные ошибки построения

выбранного вида графика по методологии data-to-viz. Если

методология Вами отвергается, то просьба обосновать Ваше решение по выбору графика.

. История должна содержать итоговые выводы. В реальных "историях о данных" именно эти выводы представляют собой основную ценность для предприятия.

Сформировать отчет и разместить его в своем репозитории на github.

Импорт библиотек

```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
import seaborn as sns
import matplotlib.pyplot as plt
sns.set(style="white",color_codes=True)
```

Описание набора данных

Набор данных Iris использовался в R.A. Классическую статью Фишера 1936 года «Использование множественных измерений в таксономических задачах» также можно найти в репозитории машинного обучения UCI.

Он включает в себя три вида ирисов по 50 образцов каждый, а также некоторые свойства каждого цветка. Один вид цветка линейно отделим от двух других, но два других не отделимы линейно друг от друга.

Столбцы в этом наборе данных:

- Id
- SepalLengthCm
- SepalWidthCm
- PetalLengthCm
- PetalWidthCm
- Species
-

Загрузка датасета

```
data = pd.read_csv("/content/Iris.csv")
print(data.head())
```

```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa
```

Шаг1.Гистограмма （Histogram）

```
data_without_id = data.drop('Id', axis=1)    # 删除名为 'ID' 的列
data_without_id.plot(kind='hist', subplots=True, layout=(2, 2), figsize=(10, 10))
```

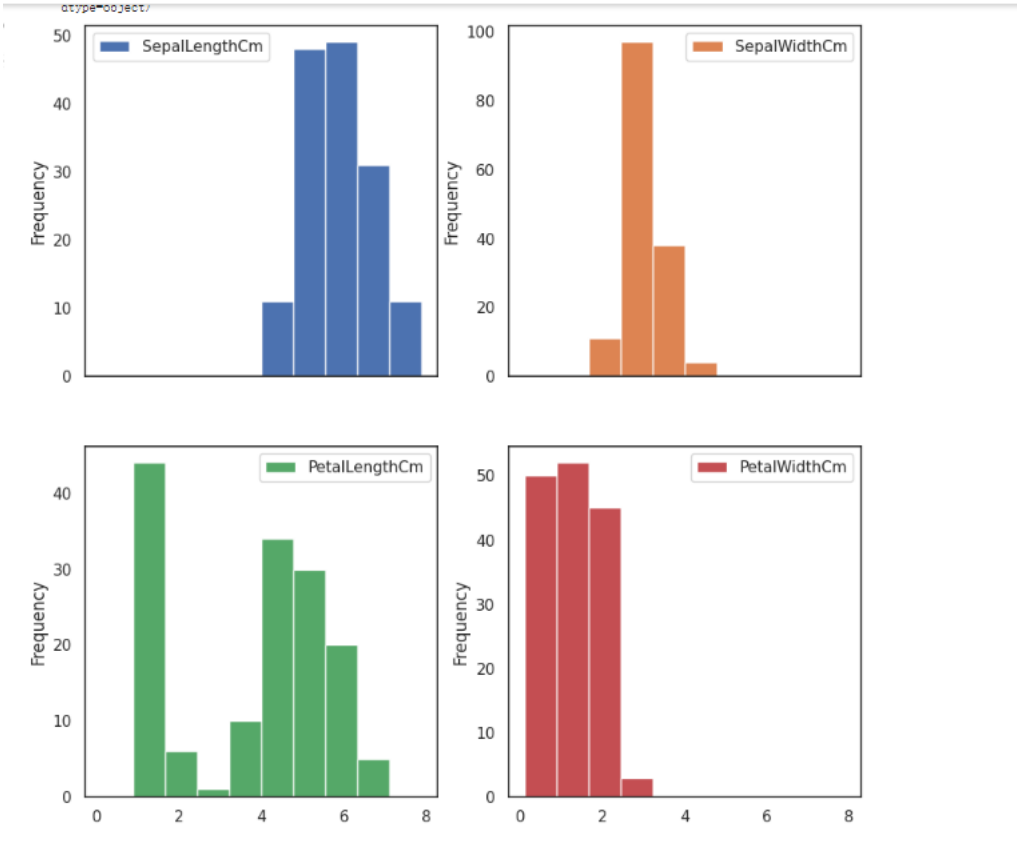


Рис- 1: Гистограмма четырех атрибутов цветка ириса

Шаг2.График рассеяния(Scatter plot)

нарисовать диаграмму рассеяния двух переменных (длина и ширина чашелистика) при построении диаграммы рассеяния.

```
#设置绘图的种类,scatter表示散点图，x轴使用萼片的长度，y轴使用萼片的宽度
data.plot(kind="scatter",x="SepalLengthCm",y="SepalWidthCm")
plt.show()
```

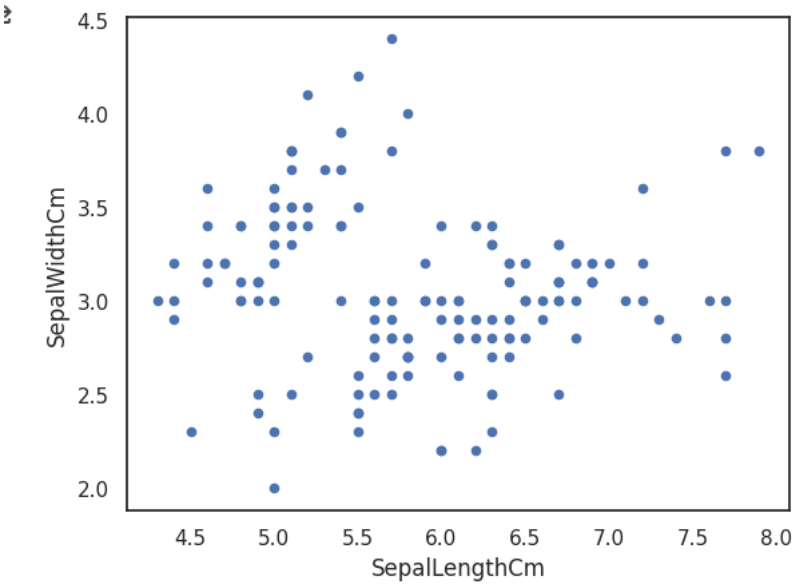


Рис- 2: неудачная график рассеяния

Используйте разные цвета, чтобы обозначить разные виды цветов ириса.

```
face  =  sns.FacetGrid(data,hue="Species",height=5)
face.map(plt.scatter,"SepalLengthCm","SepalWidthCm")
face.add_legend()
plt.show()
```

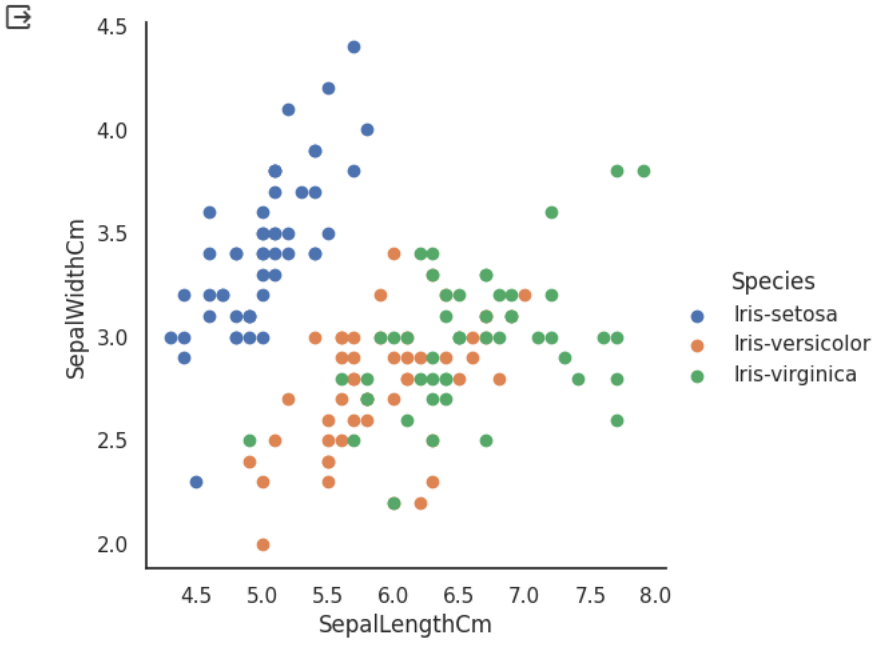


Рис- 3: удачная график рассеяния

Шаг3.коробочный сюжет(boxplot)

Нарисуйте коробчатую диаграмму в зависимости от длины цветков ириса.

```
#  箱型图
sns.boxplot(x="Species",y="SepalLengthCm",data=data)
plt.show()
```

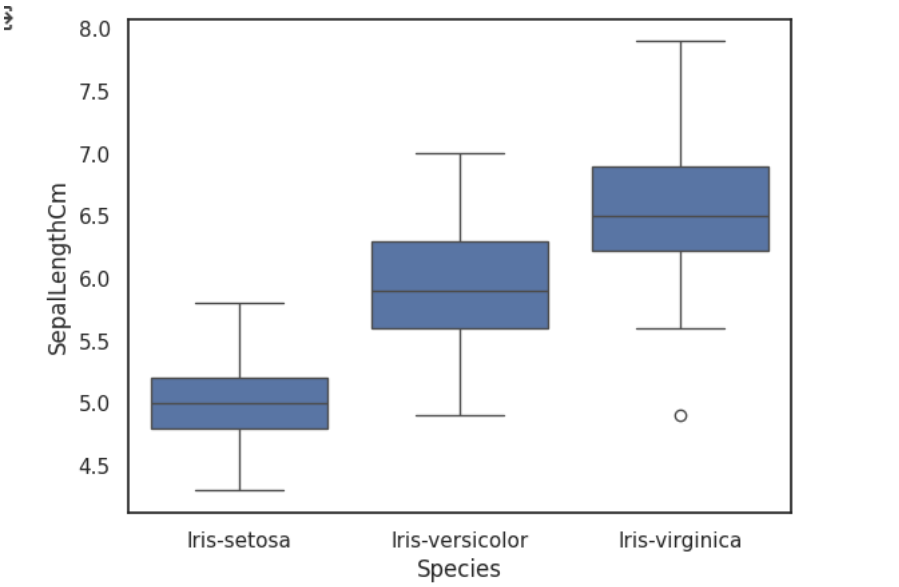


Рис- 4: неудачная коробочный сюжет

График на основе коробчатого графика:

```
ax  =  sns.boxplot(x="Species",y="SepalLengthCm",data=data)
#在箱形图的基础上进行描点，设置jitter为True保证点不会落在同一条直线上
ax  =  sns.stripplot(x="Species",y="SepalLengthCm",data=data,jitter=True,edgecolor="gray")
plt.show()
```

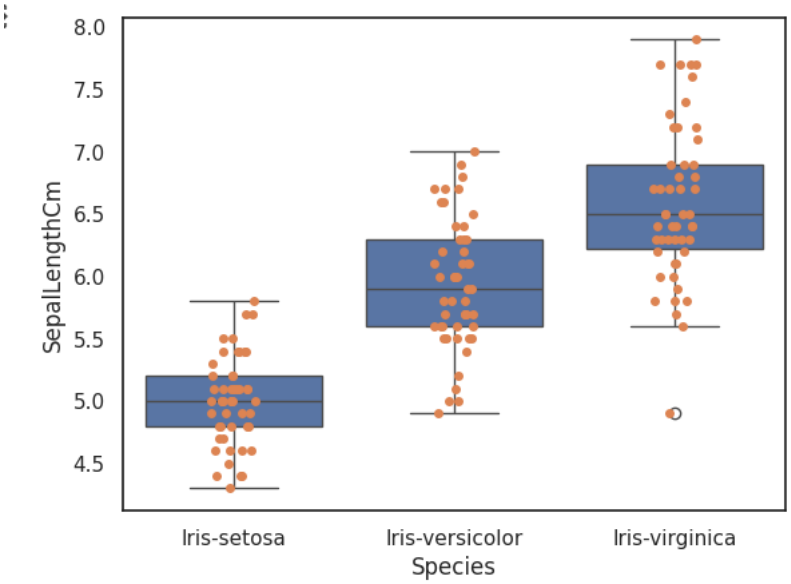


Рис- 5: удачная коробочный сюжет

Шаг 4.Нарисуйте схему скрипки（Violin diagram）

```
sns.violinplot(x="Species",y="SepalLengthCm",data=data,gridsize=6)
plt.show()
```

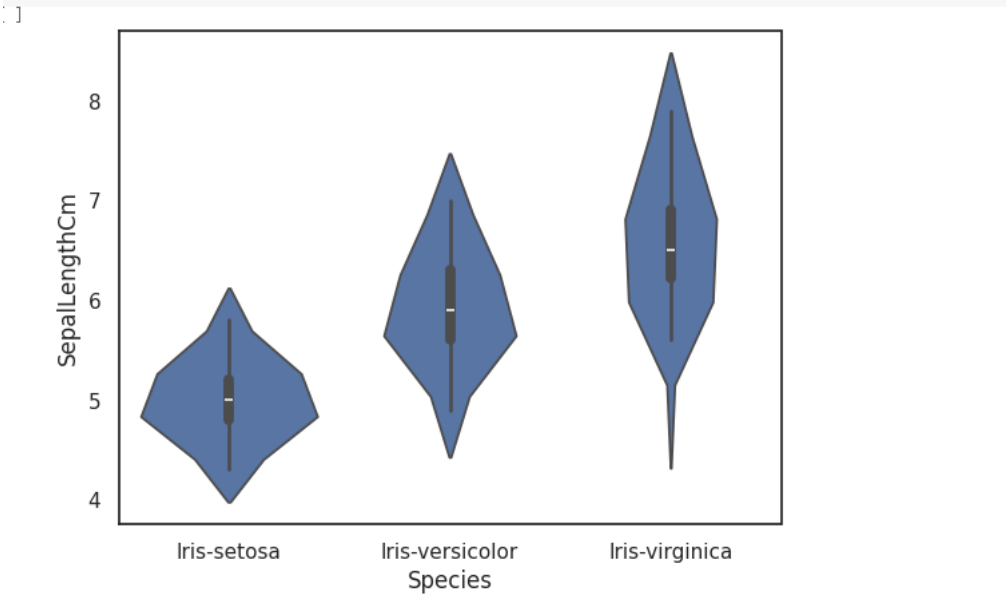


Рис- 6: Схема скрипки

Шаг 5：Оценка плотности ядра（Plot Kernel Density Estimate）

```
face = sns.FacetGrid(data,hue="Species",height=6)
face.map(sns.kdeplot,"SepalLengthCm")
face.add_legend()
plt.show()
```

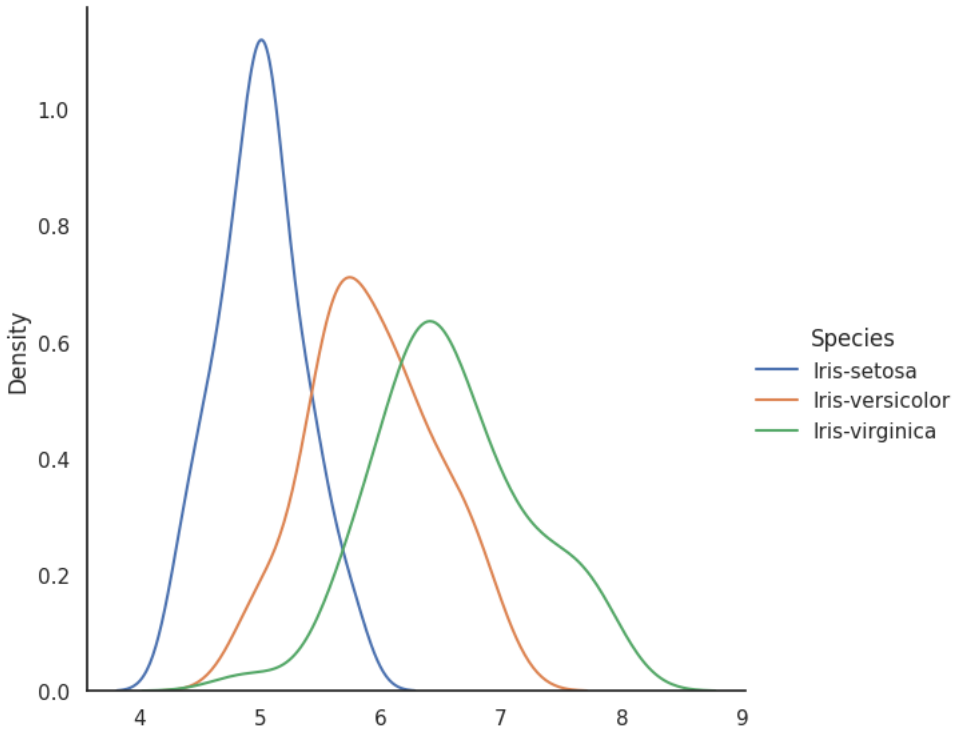


Рис- 7：График оценки плотности ядра