Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №1

по дисциплине «Методы машинного обучения»

на тему«Создание "истории о данных"»

Выполнил:

студент группы: ИУ5-23М

Аимань Мухэяти

Москва — 2020 г.

1. Цель лабораторной работы

изучение различных методов визуализация данных и создание истории на основе данных.

2. 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.

⑤ 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.

3. Ход выполнения работы

Побочные реакции вакцины (VAERS)

VAERS, система отчетности о нежелательных явлениях вакцины, введенная в действие законом 1986 года. В 2016 году VAERS получила 59 117 сообщений следующим образом: 432 смерти, 1091 постоянная инвалидность, 4132 госпитализации, 10 284 посещения отделений неотложной помощи, 59 117 общих нежелательных явлений, зарегистрированных в 2016 году.

Поскольку я также была вакцинирована против COVID-19, были и побочные реакции, но я знала, что это нормально.Поэтому я хотела бы сделать доклад о побочных реакциях на вакцину COVID-19.

Подключим все необходимые библиотеки

In [2]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
import plotly.offline as py
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

Загрузим непосредственно данные(2021vaerssymptoms.csv):

In [7]:
```
df = pd.read_csv('2021VAERSSYMPTOMS.csv', encoding='utf8')
pd.set_option('display.max_columns', None)
df.head()
```

Посмотрим на данные в данном наборс данных:

Out[7]:

	VAERS_ID	SYMPTOM1	SYMPTOMVERSION1	SYMPTOM2	SYMPTOMVERSION2	SYMPTOM3	SYMPTOMVERSION3	SYMPTOM4	SYMPTOM
0	916710	Appendicitis	23.1	Band neutrophil percentage increased	23.1	Surgery	23.1	White blood cell count increased	
1	916741	Chills	23.1	Complex regional pain syndrome	23.1	Fatigue	23.1	Headache	
2	916741	Myalgia	23.1	Pain in extremity	23.1	Peripheral swelling	23.1	X-ray abnormal	
3	916742	Anaphylactic reaction	23.1	Blood test	23.1	Burning sensation	23.1	Central venous catheterisation	
4	916742	Intensive care	23.1	Pruritus	23.1	Rash	23.1	Rash macular	

Проверяем количество пропущенных данных в каждом столбце

```
In [6]: df.isnull().sum()

Out[6]: VAERS_ID              0
        SYMPTOM1              0
        SYMPTOMVERSION1       0
        SYMPTOM2            970
        SYMPTOMVERSION2     970
        SYMPTOM3           1631
        SYMPTOMVERSION3    1631
        SYMPTOM4           2155
        SYMPTOMVERSION4    2155
        SYMPTOM5           2561
        SYMPTOMVERSION5    2561
        dtype: int64
```

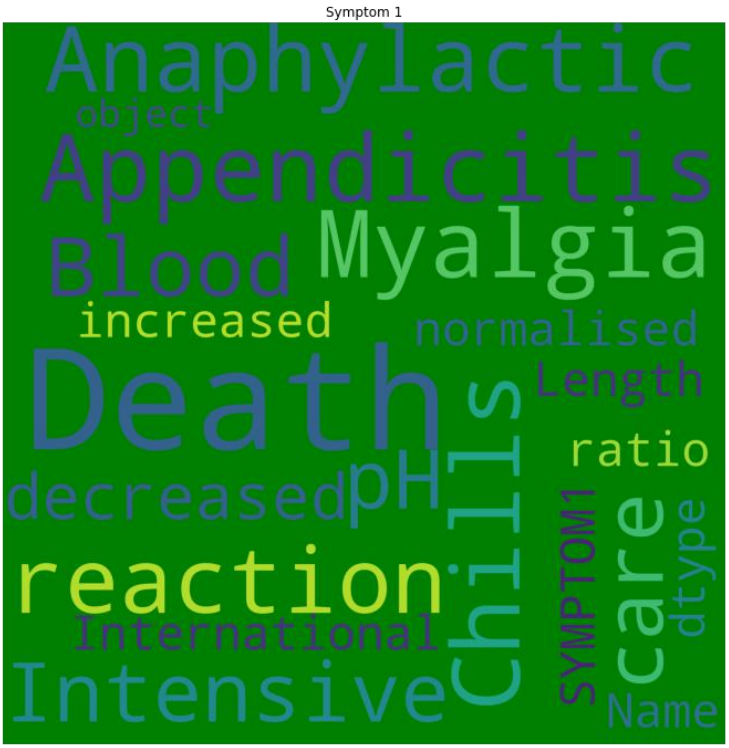
Заполняем недостающие данные с помощью режима

```
In [8]: from sklearn.impute import SimpleImputer
        imputer = SimpleImputer(strategy='most_frequent')
        df.iloc[:,:] = imputer.fit_transform(df)
```

С помощью библиотеки "облако слов" мы составляем диаграмму симптома 1. Интуитивно покажите нам, как часто появляется каждый симптом.

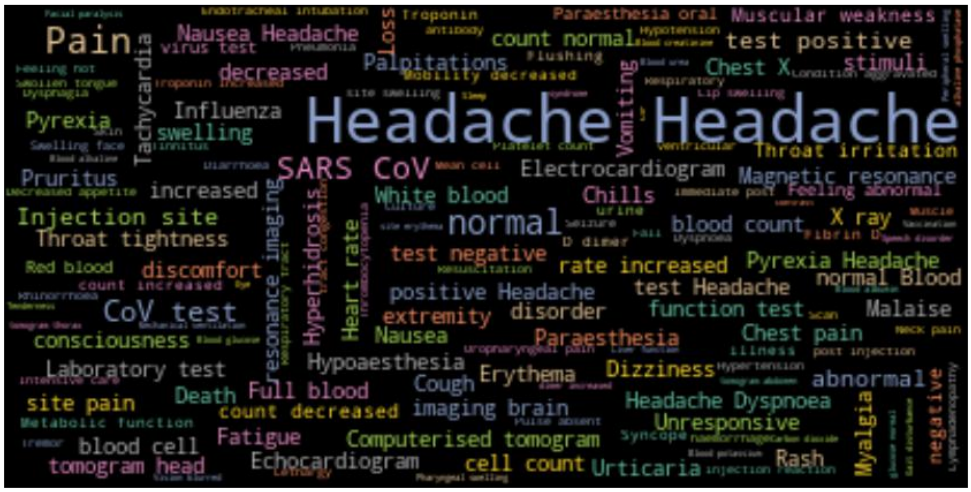
```
In [10]: from wordcloud import WordCloud
         from wordcloud import STOPWORDS
         stopwords = set(STOPWORDS)
         wordcloud = WordCloud(background_color = 'green',
                               height =2000,
                               width = 2000
                              ).generate(str(df["SYMPTOM1"]))
         plt.rcParams['figure.figsize'] = (12,12)
         plt.axis("off")
         plt.imshow(wordcloud)
         plt.title("Symptom 1")
         plt.show()
```

Мы видим самую высокую долю смертей, за которыми следуют миалгия и аппендицит



Затем мы провели аналогичную операцию над симптомами 4,2,3,5 соответственно, для создания и генерации образов облака слов

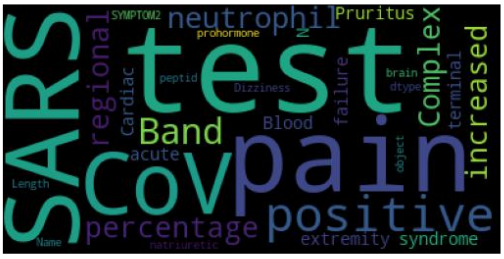
```
In [11]: from wordcloud import WordCloud, ImageColorGenerator
text = " ".join(str(each) for each in df.SYMPTOM4)
# Create and generate a word cloud image:
wordcloud = WordCloud(max_words=200,colormap='Set2', background_color="black").generate(text)
plt.figure(figsize=(10,6))
plt.figure(figsize=(15,10))
# Display the generated image:
plt.imshow(wordcloud, interpolation='Bilinear')
plt.axis("off")
plt.figure(1,figsize=(12, 12))
plt.show()
```



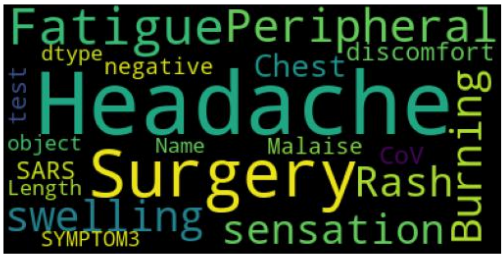
In [12]:
```
text_cols = ['SYMPTOM2', 'SYMPTOM3', 'SYMPTOM5']

from wordcloud import WordCloud, STOPWORDS

wc = WordCloud(stopwords = set(list(STOPWORDS) + ['|']), random_state = 42)
fig, axes = plt.subplots(2, 2, figsize=(20, 12))
axes = [ax for axes_row in axes for ax in axes_row]

for i, c in enumerate(text_cols):
  op = wc.generate(str(df[c]))
  _ = axes[i].imshow(op)
  _ = axes[i].set_title(c.upper(), fontsize=24)
  _ = axes[i].axis('off')

_ = fig.delaxes(axes[3])
```

Видно,что головные боли,лихорадка также являются наиболее распространенными побочными реакциями

SYMPTOM2



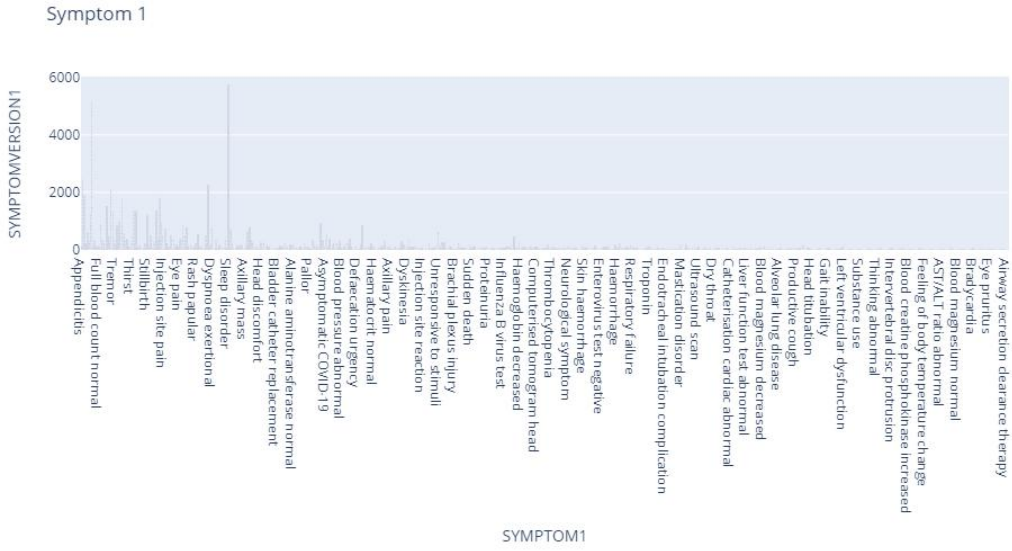
SYMPTOM3



SYMPTOM5

создаем и генерируем гистограмму симптома 1

In [13]:
```
fig = px.bar(df,
             x='SYMPTOM1', y='SYMPTOMVERSION1', color_discrete_sequence=['#2B3A67'],
             title='Symptom 1', text='VAERS_ID')
fig.show()
```



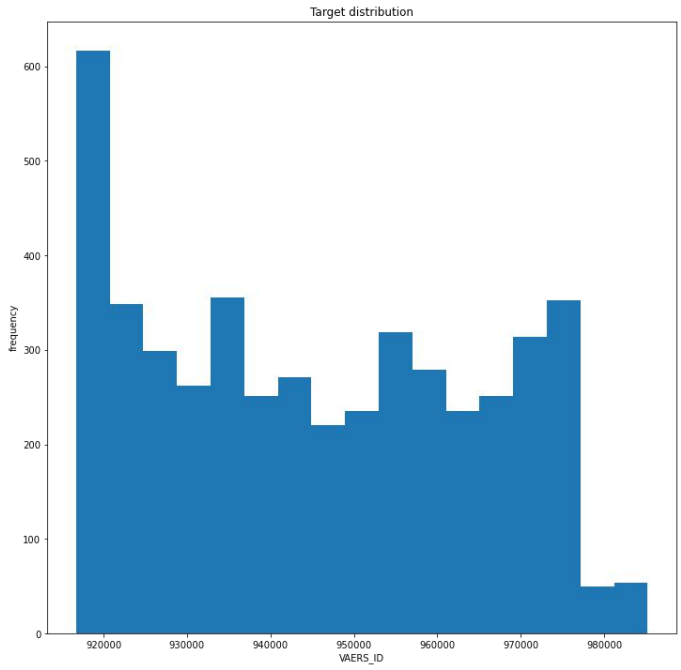
Для обработки данных используйте базовую библиотеку анализа данных.(data analysis baseline library) dabl

Исследовательский данные анализ дает мне быстрое понимание данных

In [14]:
```
! pip install -q dabl country_converter
```

In [15]:
```
import dabl
import warnings
```

In [16]:
```
#plt.style.use('classic')
new_df = df[['VAERS_ID', 'SYMPTOM1', 'SYMPTOM2', 'SYMPTOM3', 'SYMPTOM4', 'SYMPTOM5']]
dabl.plot(new_df, target_col='VAERS_ID')
```



Загрузим непосредственно данные(2021vaersvax.csv):

```
In [18]: df1 = pd.read_csv('2021VAERSVAX.csv', encoding='utf8')
         pd.set_option('display.max_columns', None)
         df1.head()
```

Посмотрим на данные в данном наборс данных:

Out[18]:

	VAERS_ID	VAX_TYPE	VAX_MANU	VAX_LOT	VAX_DOSE_SERIES	VAX_ROUTE	VAX_SITE	VAX_NAME
0	916710	COVID19	MODERNA	NaN	1	IM	LA	COVID19 (COVID19 (MODERNA))
1	916741	COVID19	PFIZER\BIONTECH	EH9899	1	SYR	LA	COVID19 (COVID19 (PFIZER-BIONTECH))
2	916742	COVID19	PFIZER\BIONTECH	NaN	1	IM	NaN	COVID19 (COVID19 (PFIZER-BIONTECH))
3	916746	COVID19	MODERNA	037K20A	1	IM	LA	COVID19 (COVID19 (MODERNA))
4	916772	COVID19	PFIZER\BIONTECH	EJ1685	UNK	IM	LA	COVID19 (COVID19 (PFIZER-BIONTECH))

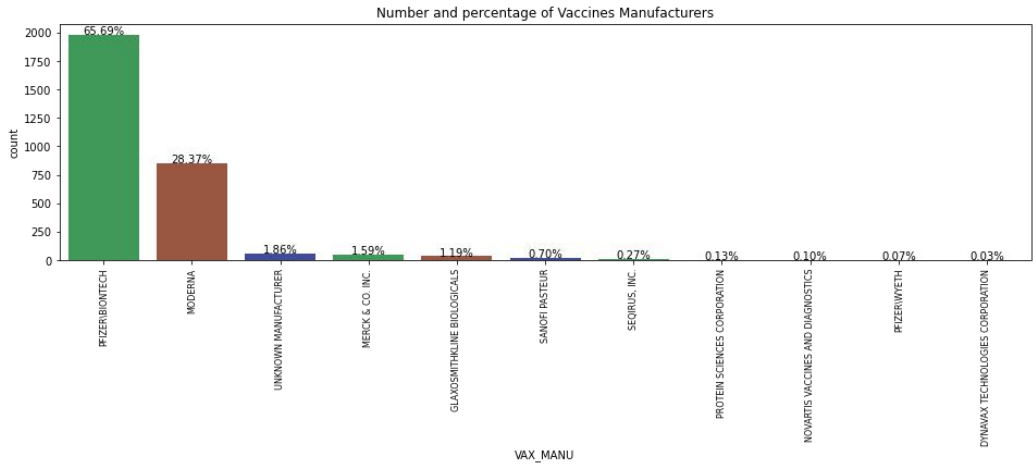
Определение функции

Количество отдельных производителей вакцин и их процентное соотношение,и в -изуализируем эту информацию

```
In [19]: def plot_count(feature, title, df, size=1):
             f, ax = plt.subplots(1,1, figsize=(4*size,4))
             total = float(len(df1))
             g = sns.countplot(df1[feature], order = df1[feature].value_counts().index[:20], palette= ('#32a852', '#a84e32', '#3242a8'))
             g.set_title("Number and percentage of {}".format(title))
             if(size > 2):
                 plt.xticks(rotation=90, size=8)
             for p in ax.patches:
                 height = p.get_height()
                 ax.text(p.get_x()+p.get_width()/2.,
                         height + 3,
                         '{:1.2f}%'.format(100*height/total),
                         ha="center")
             plt.show()
```

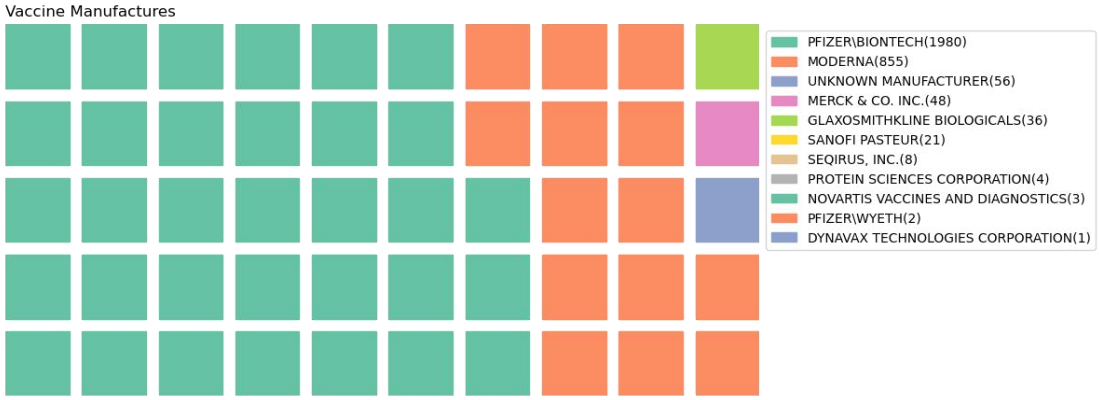
```
In [20]: plot_count("VAX_MANU", "Vaccines Manufacturers", df1, 4)
```

Как видно,Производитель вакцин PFIZER/ BIONTECH имеет самые неблагоприятные реакции на вакцину.За ним следует производитель вакцин MODERNA.Остальные производители вакцин составляли небольшую долю



Другой способ показать, теперь мы создаем и генерируем "Вафельные" изображения

```
In [34]: from pywaffle import Waffle

plt.figure(
    FigureClass=Waffle,
    rows=5,
    columns=10,
    values=variable,
    title={'label': 'Vaccine Manufactures', 'loc': 'left'},
    labels=["{} ({})".format(a, b) for a, b in zip(variable.index, variable) ],
    # Set the position of the legend
    legend={'loc': 'upper left', 'bbox_to_anchor': (1, 1)},
    dpi=100
)
plt.show()
```



Загрузим непосредственно данные(2021vaersdata.csv):

```
In [37]: patient = pd.read_csv('2021VAERSDATA.csv', encoding='windows-1254')

In [39]: new_patient_data = patient.merge(df1, on="VAERS_ID")

In [40]: new_patient_data['RECOVD'] = new_patient_data['RECOVD'].replace(np.nan, 'Unknown')
         new_patient_data['RECOVD'] = new_patient_data['RECOVD'].replace('Y', 'Yes')
         new_patient_data['RECOVD'] = new_patient_data['RECOVD'].replace('N', 'No')
         new_patient_data['RECOVD'] = new_patient_data['RECOVD'].replace('U', 'Unknown')

In [41]: fig = px.scatter(new_patient_data, x="AGE_YRS", y="VAERS_ID", color="SEX")
         fig.update_layout(
             title_text='Age Distribution and Gender',
             xaxis_title_text='Age',
             yaxis_title_text='ID'
         )
         fig.show()
```

В этой таблице мы видим распределение по возрасту и полу.Синий для женщин и красный для мужчин.Зеленый для неизвестного

Есть больше женщин с побочными реакциями и больше мужчин старше 60 лет.

Этот график представляет собой распределение распределения по возрасту и состоянию восстановления.

```
In [42]: fig = px.scatter(new_patient_data, x="AGE_YRS", y="VAERS_ID", color="RECOVD")
         fig.update_layout(
             title_text='Age Distribution and Recovery Status',
             xaxis_title_text='Age',
             yaxis_title_text='ID'
         )
         fig.show()
```

Красный означает отсутствие восстановления,зеленый означает восстановление.Синий цвет символизирует неизвестность.Видно, что общее частичное выздоровление не больше, а население старше 60 лет выздоравливает намного меньше.

Хотим знать распределение иммунных путей в организме человека для каждой в акцины.

In [45]:
```
r = new_patient_data['VAX_ROUTE'].value_counts()
route = pd.DataFrame({'Route':r.keys(), 'Count': r.values})
route_fig = px.pie(route, values="Count", names="Route", title="Vaccination Route to Human Body")
route_fig.update_layout(
    title_text='Vaccination Route to Human Body',
    xaxis_title_text='Route',
    yaxis_title_text='Count',
)
route_fig.show()
```

Видно, что IM иммунный путь Замби больше всего, за ним следует OT.

Эта круго-вая диаграмма показывает долю и общее количество иммунных путей

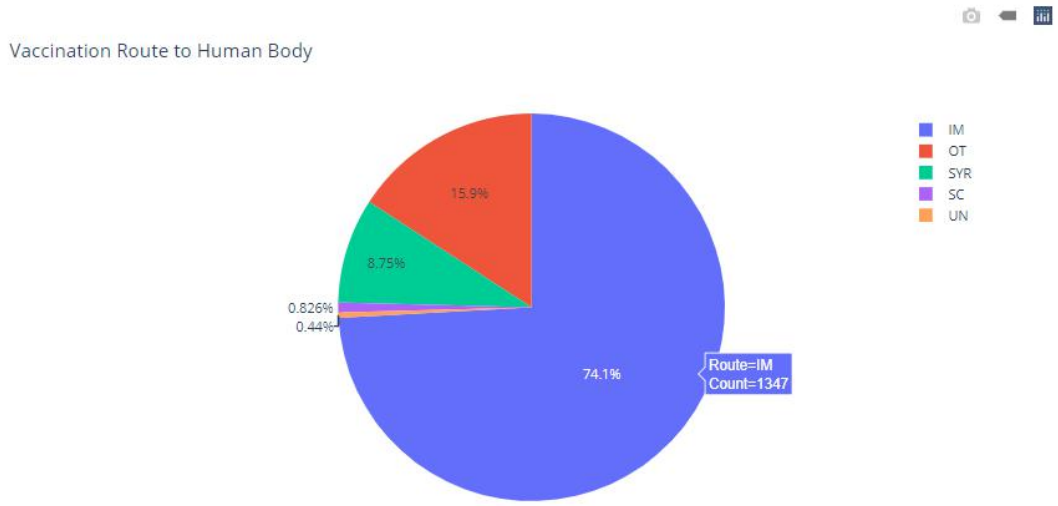
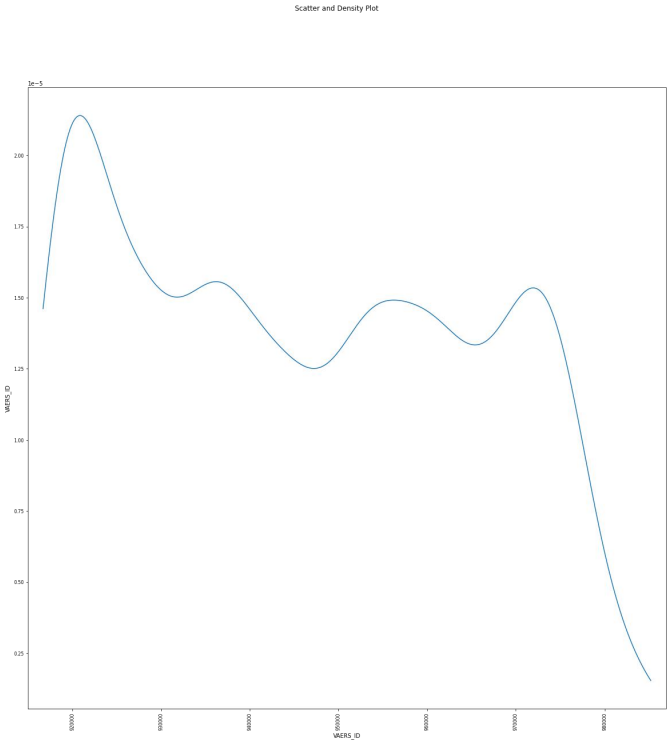


Диаграмма рассеяния и карты плотности для каждой вакцины

In [51]:
```
def plotScatterMatrix(df, plotSize, textSize):
    df = df.select_dtypes(include =[np.number]) # keep only numerical columns
    # Remove rows and columns that would lead to df being singular
    df = df.dropna('columns')
    df = df[[col for col in df if df[col].nunique() > 1]] # keep columns where there are more than 1 unique values
    columnNames = list(df)
    if len(columnNames) > 10: # reduce the number of columns for matrix inversion of kernel density plots
        columnNames = columnNames[:10]
    df = df[columnNames]
    ax = pd.plotting.scatter_matrix(df, alpha=0.75, figsize=[plotSize, plotSize], diagonal='kde')
    corrs = df.corr().values
    for i, j in zip(*plt.np.triu_indices_from(ax, k = 1)):
        ax[i, j].annotate('Corr. coef = %.3f' % corrs[i, j], (0.8, 0.2), xycoords='axes fraction', ha='center', va='center', s
    plt.suptitle('Scatter and Density Plot')
    plt.show()
```



Вакцины могут вызывать побочные реакции, но побочные реакции делятся на различные степени и различные уровни, как правило, легкие, умеренные и тяжелые.В целом вакцины COVID-19 сейчас на рынке очень много.Среди производителей вакцин доминирует компания Rycon.Побочные реакции различаются по возрасту и полу.Женщины и пожилые люди более склонны к побочным реакциям, а пожилые вакцинаторы не так легко восстанавливаются после побочных реакций.

Список литературы

[1] Гапанюк Ю. Е. Лабораторная работа «истории о данных» [Электронный ресурс] // GitHub. — 2021. — Режим доступа: https://github.com/ugapanyuk/ml_course_2021/wiki/COURSE_MMO