**Прогнозирование осадков с помощью  
машинного обучения**

( Соловьева Анастасия Федоровна МИК21 )

#### **Введение**

В данном проекте была проведена работа по предсказанию осадков на основе исторических данных, используя методы машинного обучения. Основной целью было создание модели, способной классифицировать наличие или отсутствие осадков, а также оценка ее производительности.   
Для достижения этой цели использовались различные алгоритмы, такие как логистическая регрессия, XGBoost и метод опорных векторов (SVC). Процесс включал в себя сбор данных, их предварительную обработку, обучение моделей и оценку их эффективности.

#### **Основные этапы работы**

**1. Загрузка и исследование данных:**

Начинаем с импорта необходимых библиотек, включая numpy, pandas, matplotlib, seaborn, а также библиотеки для машинного обучения из scikit-learn и xgboost. Здесь загружается CSV-файл с данными о осадках от data.gov.

После загрузки данные отображаются с помощью df.head(), чтобы увидеть первые несколько строк.

Код ввода данных:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sb

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn import metrics

from sklearn.svm import SVC

from xgboost import XGBClassifier

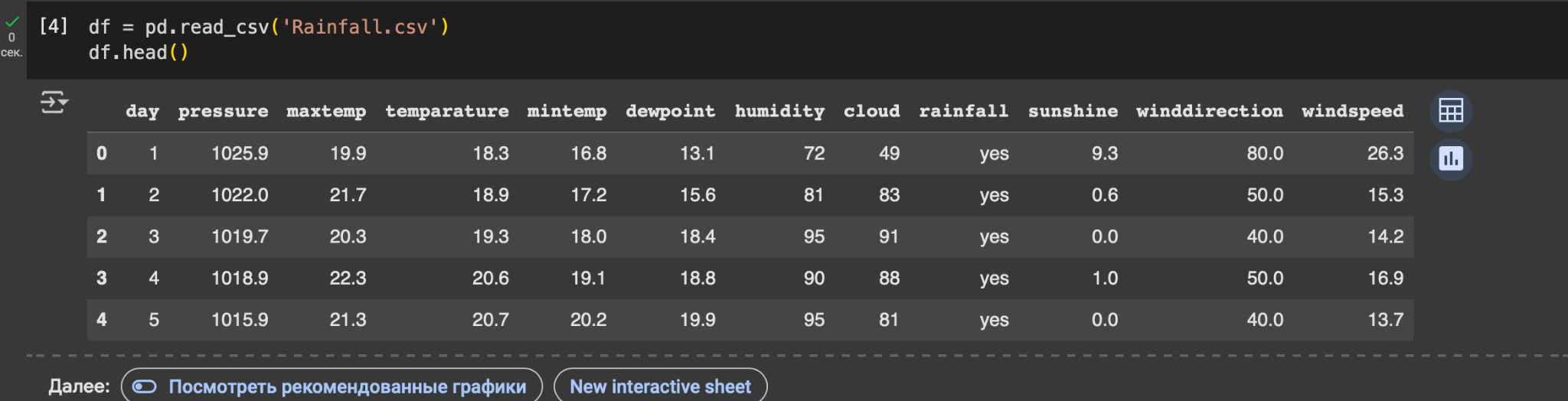
from sklearn.linear\_model import LogisticRegression

from imblearn.over\_sampling import RandomOverSampler

import warnings

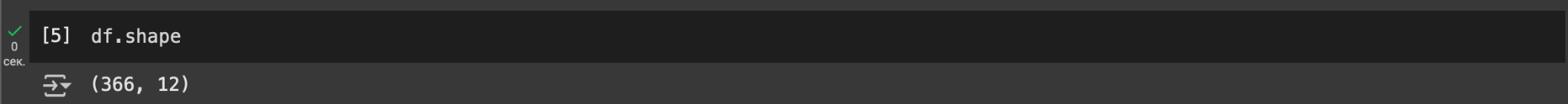
warnings.filterwarnings('ignore')

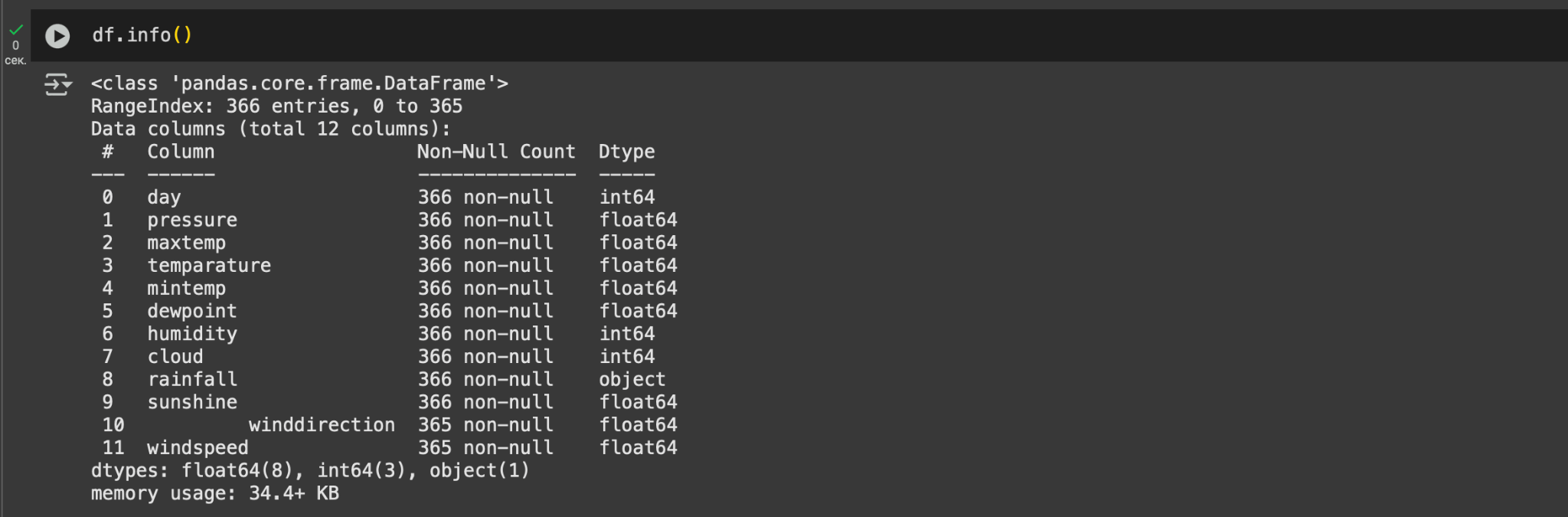
Первые несколько строк:



Исследуем форму данных (df.shape), их типы и статистику (df.info(), df.describe().T). Также проверяется наличие пропусков в данных (df.isnull().sum()).

Выходные данные:







Согласно приведенной выше информации о данных в каждом столбце, мы можем заметить, что нет нулевых значений.

**2. Очистка данных**

Пропуски в числовых столбцах были заполнены средними значениями, что позволило сохранить целостность данных. Так же при очистки данных были удалены лишние пробелы.

Код для заполнения средними значениями:

for col in df.columns:

# Checking if the column contains

# any null values

if df[col].isnull().sum() > 0:

val = df[col].mean()

df[col] = df[col].fillna(val)

df.isnull().sum().sum()

Код для лишних пробелов:

df.rename(str.strip,

axis='columns',

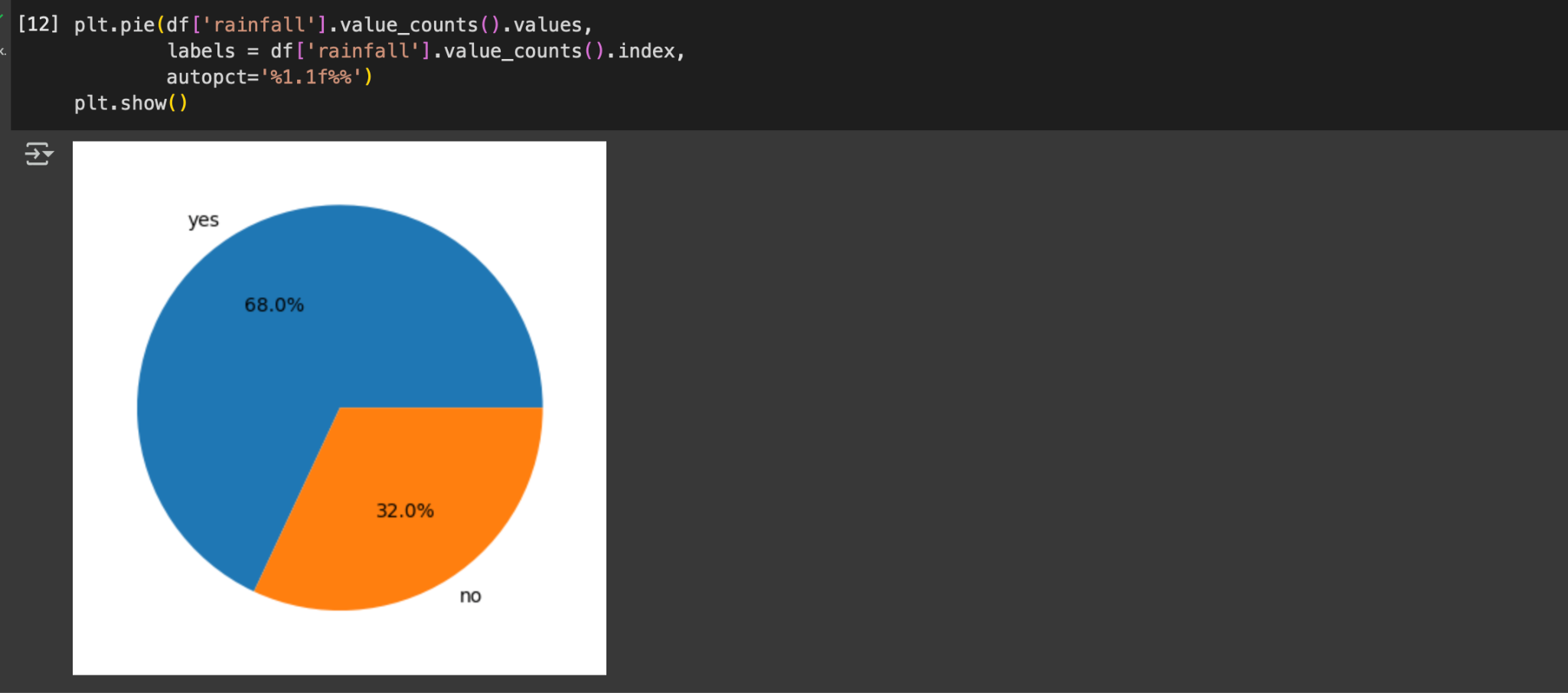
inplace=True)

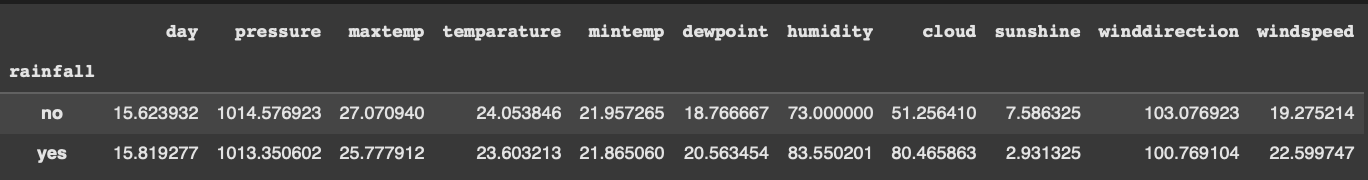
df.columns

**4. Анализируем данные**

После того, как почистили все данные, используем EDA - это подход к анализу данных с использованием визуальных методов. Он используется для обнаружения тенденций и закономерностей или для проверки предположений с помощью статистических резюме и графических представлений.

Здесь мы увидим, как проверить дисбаланс данных и перекос данных.



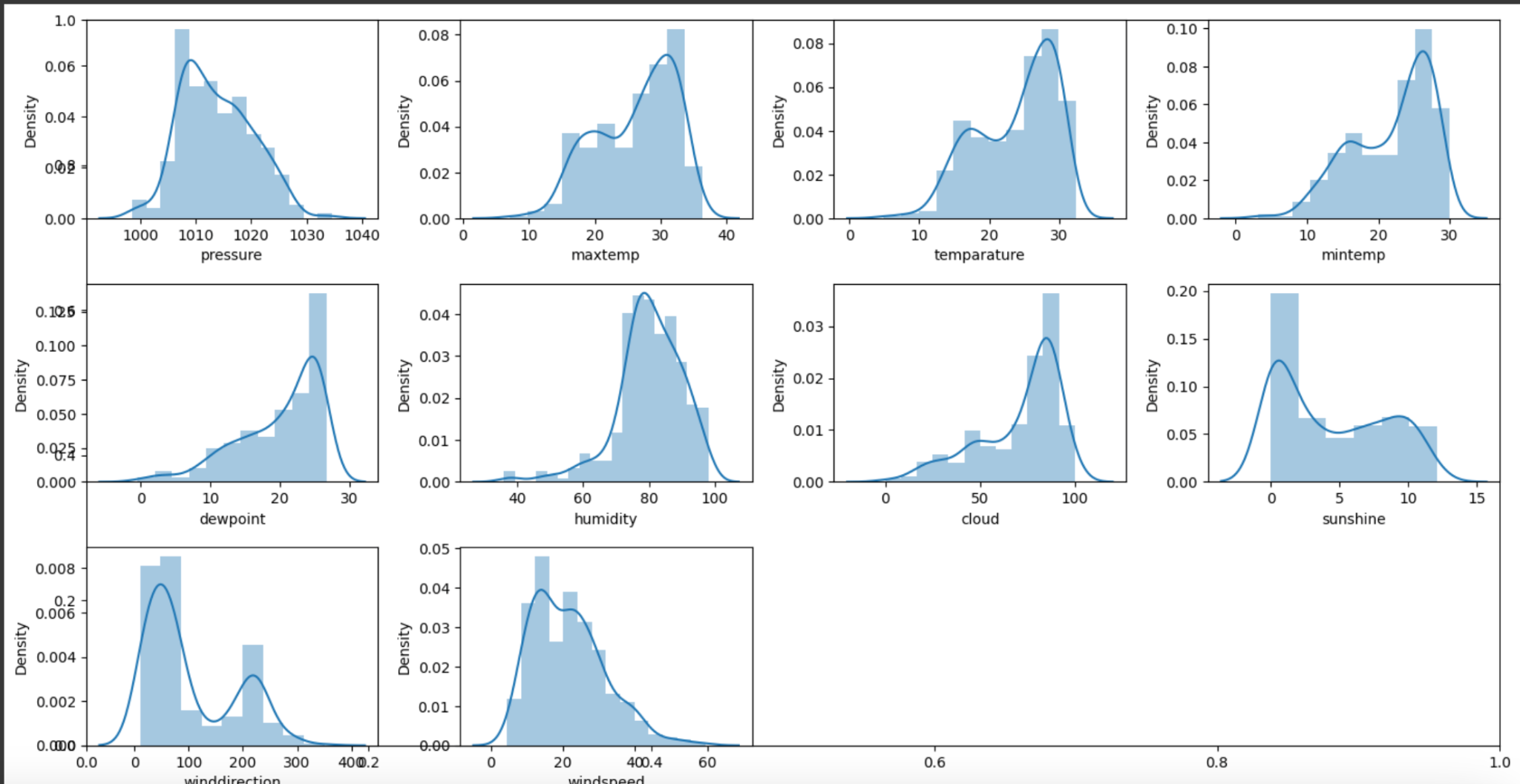


Здесь мы можем четко провести некоторые наблюдения:

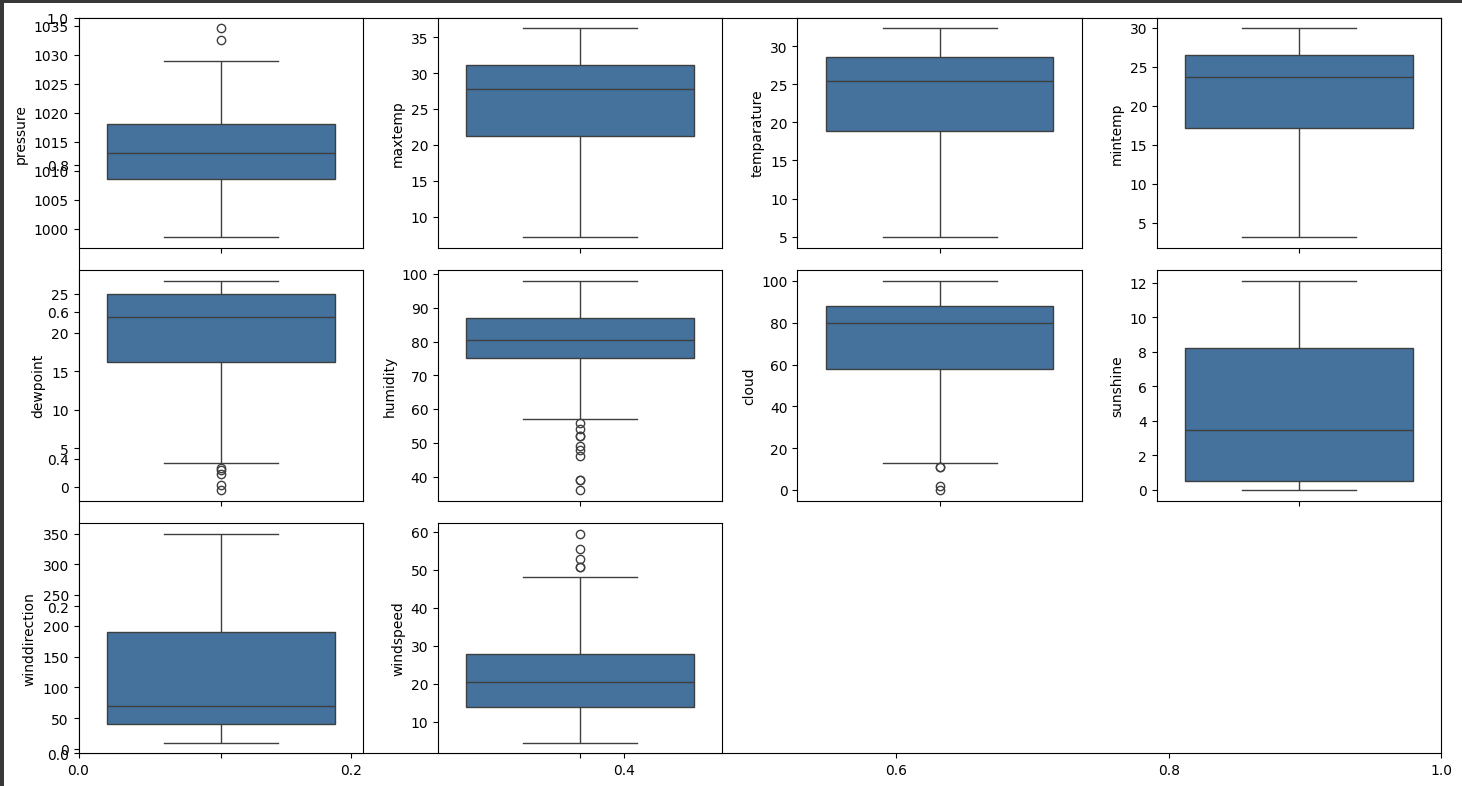
* максимальная температура относительно ниже в дни осадков
* значение точки росы выше в дни осадков
* влажность высокая в дни, когда ожидается дождь
* очевидно, что там должны быть облака для осадков
* солнечного света также меньше в дни дождей
* скорость ветра выше в дни дождей

Наблюдения, которые мы взяли из вышеуказанного набора данных, очень похожи на то, что наблюдается и в реальной жизни.

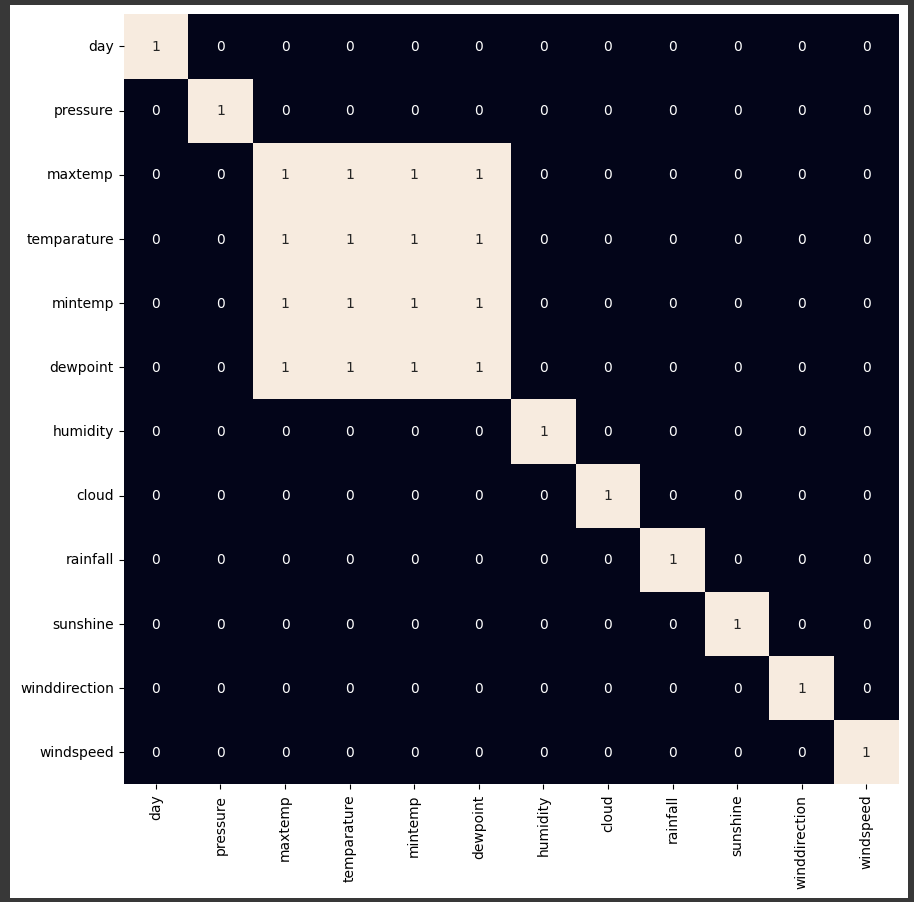
Здесь проверим распределение непрерывных функций, приведенных в наборе данных.



Нарисуем boxplots для непрерывной переменной, чтобы обнаружить отложения, присутствующие в данных. ( В наших данных есть отходы, но, к сожалению, у нас не так много данных, поэтому мы не можем их удалить )



Иногда существуют высоко коррелированные особенности, которые просто увеличивают размерность пространства функций и не полезны для производительности модели. Поэтому мы должны проверить, есть ли в этом наборе данных высоко коррелированные функции или нет.



**6. Обучение моделей**

Код обучает три разные модели: логистическую регрессию, XGBoost и SVC (Support Vector Classifier). Для каждой модели вычисляется точность на обучающей и валидационной выборках:

Код для обучения:

models = [LogisticRegression(), XGBClassifier(), SVC(kernel='rbf', probability=True)]

for i in range(3):

models[i].fit(X, Y)

print(f'{models[i]} : ')

train\_preds = models[i].predict\_proba(X)

print('Training Accuracy : ', metrics.roc\_auc\_score(Y, train\_preds[:,1]))

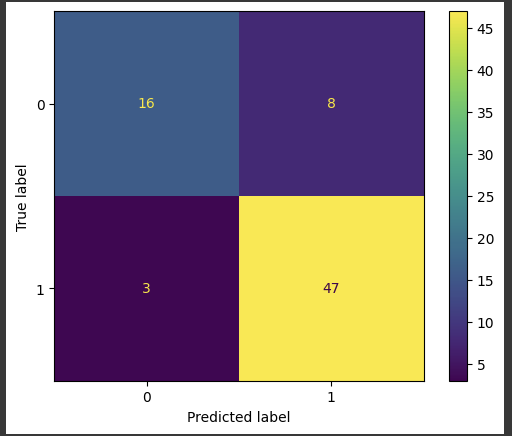
val\_preds = models[i].predict\_proba(X\_val)

print('Validation Accuracy : ', metrics.roc\_auc\_score(Y\_val, val\_preds[:,1]))

print()

**7. Оценка производительности**

Для одной из моделей (в данном случае SVC) выводится матрица путаницы и отчет о классификации:



#### **Вывод**

#### В результате проведенного анализа и обучения моделей удалось создать эффективную систему предсказания осадков. Сравнение различных алгоритмов показало, что каждая модель имеет свои сильные и слабые стороны, что подчеркивает важность выбора подходящего метода в зависимости от специфики задачи. Полученные результаты могут быть использованы для дальнейшего улучшения моделей и более точного предсказания осадков в будущем. Проект продемонстрировал значимость предварительной обработки данных и оценки моделей в процессе машинного обучения.