СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc121166877)

[1 Выбор и подгтовка модели машинного обучения 5](#_Toc121166878)

1.1 Выбор модели и писание данных………………..……………………5

1.2 Сериализация выбранной модели …………………………………….6

[2 Разработка дашборда](#_Toc121166882) 7

[2.1 Разработка страницы с описанием датасета](#_Toc121166883) 7

[2.2 Разработка страницы с показателем точности модели 7](#_Toc121166884)

[2.3 Разработка страницы для ввода пользовательского примера 8](#_Toc121166885)

[2.4 Разработка дополнительных функций, необходимых для работы дашборда 8](#_Toc121166885)

[ЗАКЛЮЧЕНИЕ](#_Toc121166892) 9

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ](#_Toc121166893) 10

ВВЕДЕНИЕ

Жизнь современного человека сложно представить без цифровой информации, ведь мы все оставляем свой цифровой след в интернете, и с каждым годом этих данных становится всё больше, ведь цифровизация охватывает всё больше и больше сфер нашей жизни. Все эти данные могут принести большую пользу человечеству, ведь их анализ может дать нам ценнейшие знания по тому, как улучшить и оптимизировать те или иные сферы нашей жизни. А машинное обучение может помочь нам в этом, ведь с его помощью мы можем выявлять сложнейшие зависимости, о которых даже не могли подумать, а также строить предсказания на будущие события. Только представьте, какие возможности это открывает перед нами.

В анализе данных и машинном обучении на языке *Python* нам могут помочь следующие библиотеки, которые были изучены в ходе курса:

1. *NumPy* – библиотека содержащая множество быстрых универсальных функций для рутинных вычислений, которые сложно делать вручную. Здесь реализовано множество вычислительных механизмов, пакет поддерживает специализированные структуры данных, в том числе — одномерные и многомерные массивы, значительно расширяющие возможности Python по выполнению различных вычислений.
2. *SciPy* – библиотека содержащая функции, которые полезны в математике и других науках. Например: статистические функции, функции оптимизации, обработки сигналов, решения дифференциальных уравнений, а также функции для нахождения численного решения интегралов.
3. *Pandas* – одна из самых популярных библиотек для анализа данных. Она предоставляет множество полезных инструментов для сбора, очистки и моделирования данных. С Pandas вы можете загружать, подготавливать, анализировать и манипулировать любыми индексированными данными. Библиотеки для машинного обучения также используют датафреймы из *Pandas* в качестве входных данных.
4. *sklearn* – один из наиболее широко используемых пакетов *Python* для анализа данных и машинного обучения, который предоставляет алгоритмы и функции для предобработки данных, уменьшения размерности, регрессии, классификации, кластерного анализа.
5. *Tensorflow* – библиотека машинного обучения от google, позволяющая работать с моделями глубокого обучения. На основе данной библиотеки строятся более высокоуровневые библиотеки для работы с нейронными сетями на уровне целых слоев.
6. *Keras* – *api* для взаимодействия с искуственными нейронными сетями, поддерживающий основные виды слоев и структурные элементы. Использует в качестве *backend* для вычислений *Tensorflow*.
7. *Pickle* – стандартный пакет языка *Python* для сериализации/десериализации объектов.

1 Выбор и подготовка модели машинного обучения

1.1 Выбор модели и описание данных

Для анализа данных был выбран датасет под названием «Rain in Australia», представленный в формате *.csv*, размещённый на онлайн-сервисе *Kaggle*. Ниже представлено краткое описание датасета.



Рисунок 1 – Описание датасета

Следующим шагом был выбор модели, на данных были протестированы множество моделей, от классических, например knn, до полносвязной нейронной сети. Всем моделям были подобраны гиперпараметры исходя из проб на тестовой части выборки, после этого качество моделей было измерено на основе метрик, таких как: accuracy, precision, recall, f1-score и confusion matrix.

Исходя из анализа показателей моделей, лучшей была выбрана модель логистической регрессии, так как она показала не только лучшее качество, но и высокую скорость обучения, так как модель имеет относительно небольшую сложность.

**1.2 Сереализация выбранной модели**

После подбора модели остаётся последняя задача, подготовка обученной модели к её переносу на дашборд. Эта операция проводилась с помощью стандартного пакета pickle, который позволяет записывать сложные объекты из оперативной памяти на внешние носители. Производится это следующим образом:

import pickle

pickle.dump(lr, open('/content/drive/MyDrive/log\_regress.sav', 'wb'))

2 Разаработка дашборда

Для удобного представления проделанной работы и возможности тестирования модели пользователями, не имеющими навыки программирования, была поставлена задача разработать дашборд. Для реализации задачи был выбран веб-фреймворк *Streamlit*, предназначенный для простого развертывания моделей и визуализаций.

2.1 Разработка страницы с описанием датасета

При помощи методов *markdown* и *header* была создана стартовая страница с описанием датасета и задачи модели машинного обучения. Ниже приведён пример кода.

**st**.header("Описание задачи")

**st**.markdown("""Задача данной модели машинного обучения – предсказание дождей в Австралии""")

2.2 Разработка страницы с показателем точности модели

При помощи методов *write* и *header*, и метода модели логистической регрессии *score* была создана страница, способная по запросу пользователя показать ему точность модели машинного обучения. Ниже приведён пример кода.

**st**.header("Точность модели")

accuracy = model.score(test\_data\_X, test\_data\_y)

**st**.write(f"{accuracy}")

2.3 Разработка страницы для ввода пользовательского примера

При помощи виджетов ввода, представленных в *Streamlit*, таких как *number\_input*, *button* и *selectbox*, была создана страница, позволяющая пользователю ввести свои данные и получить по ним предсказание. Ниже приведены примеры использования вышеперечисленных виджетов.

 dt["Cloud9am"] = **st**.number\_input("Облачность в 9 утра", 1, 8)

 if **st**.button('Предсказать'):

                data = **list**(dt.values())

                data = [**int**(str) for str in data]

                data = **np**.**array**(data).**reshape**((1, -1))

                pred = model.predict(data)

**st**.write(f"Предсказанное значение: {pred[0]:.2f}")

wind\_gust\_dir = **st**.selectbox("Направление самого сильного порыва ветра", ['W', 'WNW', 'WSW', 'NE', 'NNW', 'N', 'NNE', 'SW','ENE',

       'SSE', 'S', 'NW', 'SE', 'ESE', 'E', 'SSW'])

2.4 Разработка дополнительных функций, необходимых для работы дашборда

Для корректной работы дашборда, ему необходимы функции для загрузки и предобработки датасета, а также для десериализации модели машинного обучения. Функции реализованы при помощи декоратора *cache\_data*, позволяющего оптимизировать вызовы функций. Пример разработанной функции для загрузки модели приведён ниже.

**@st.cache\_data**

def **load\_model**(path\_to\_file):

    with **open**(path\_to\_file, 'rb') as model\_file:

        model = **pickle**.**load**(model\_file)

    return model

ЗАКЛЮЧЕНИЕ

В ходе выполнения расчётно-графической работы были выполнены следующие задачи:

# Проведена предобработка данных их подробное описание.

# Проведено обучение множества моделей с подбором гиперпараметров и оценкой их точности.

# На основе метрик была выбрана лучшая модель.

# Для выбранной модели был разработан дашборд.

# В результате работы над расчетно-графической работой удалось выполнить все поставленные задачи.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

# Scipy documentation. URL: https://docs.scipy.org/doc/scipy.

# Matplotlib documentation. URL: https://matplotlib.org/stable/index.html.

# PyData. Pandas documentation. URL: https://pandas.pydata.org/docs.

# Scikit-learn documentation. URL: https://scikit-learn.org/stable/index.html.