|  |  |  |
| --- | --- | --- |
| Category | Description | Points |
| 1. Избор на алгоритъм | Избраният алгоритъм е подходящ за решаването на проблема | 10 |
| 2. Моделът е имплементиран изцяло от ученика | не са използвани библиотеки като Sci-kit learn, keras, tensorflow, pytorch. Позволено е да използвате pandas, numpy, matplotlib, joblib и други подобни.  Tensorflow и pytorch могат да бъдат използвани само за създаване на масиви/тензори, които могат да се смятат с графична карта.  пример:  **Pytorch**: self.weights = torch.randn(input\_size, 1, device="cuda", dtype=torch.float32, requires\_grad=True)  **Tensorflow**: self.w = tf.Variable(tf.random.normal([input\_dim, 1]), name='weights')  В случай, че е използвана библиотека за изпълнението на алгоритъма, максималният допустим резултат за тази графа е 10 точки. | 20 |
| 3. Датасет | писмено обяснение и история от къде е взет/генериран/събран датасетa и как е бил преработен. (свободен текст) | 10 |
| 5. Ефективността на модела се следи | да се проучат и използват метриките accuracy, error, loss, както и други при необходимост | 10 |
| 6. Избраните признаци(features) са основателни | признаците, използвани за обучението на алгоритъма, са добре обмислени, след което тествани и най-накрая избрани. | 10 |
| 7. Интеграция в уеб приложение | Моделът е част от уебсайт и генерира предсказание спрямо входовете на потребителя | 10 |
| TOTAL |  | 100 |

# Изисквания за сайт

1. **Технически Изисквания**

|  |  |  |
| --- | --- | --- |
| № | Изискване | Точки |
| 1 | Проекта е разработен с Python и Flask | 5 |
| 2 | Следните разширения на Flask трябва да бъдат използвани и интегрирани по подходящ начин: Flask-Bootstrap, Flask-WTF, Flask-SQLAlchemy, Flask-Login, Flask-Migrate, Flask-Mail. | 20 |
| 3 | Да се използват Flask Blueprints за структуриране на проекта на модули. | 10 |
| 4 | Да се използва механизъм за потвърждение на имейл чрез токенизация (например чрез Flask-Mail и itsdangerous). | 5 |
| 5 | Да се създадат и изпълнят unit тестове за ключови функционалности. | 10 |
| 6 | Да се създадат персонализирани страници за грешки (напр. 404, 500), които да се зареждат автоматично при възникване на грешка. | 5 |
| 7 | Сайтът трябва да включва интерфейс за попълване на анкети от потребителите, като събраните данни се използват за обучение на AI модел. | 20 |
| 8 | Интерфейсът на сайта трябва да позволява на потребителите да взаимодействат с обучената AI функционалност (напр. прогнозиране или класификация). | 25 |
|  | total | 100 |

1. **Функционални Изисквания**

|  |  |  |
| --- | --- | --- |
| № | Изискване | Точки |
| 1 | Системата трябва да поддържа регистрация, вход и изход на потребители с използване на Flask-Login. | 10 |
| 2 | Системата трябва да поддържа различни потребителски роли – поне „потребител“ и „администратор“. | 10 |
| 3 | Всеки потребител трябва да разполага със собствен профил и да може да редактира своите лични данни. | 15 |
| 4 | Потребителите трябва да могат да виждат резултатите от AI прогнозите на други потребители, с тяхно съгласие. | 20 |
| 5 | Администраторите трябва да могат да редактират или изтриват потребителски профили. | 15 |
| 6 | Интерфейсът трябва да позволява лесен достъп до функционалностите, свързани с попълване на анкети и използване на AI | 10 |
| 7 | Сайтът трябва да включва потребителски интерфейс за преглед и анализ на резултатите от обучените AI модели. | 10 |
| 8 | Навигацията и потребителският интерфейс са интуитивни | 10 |
|  | total | 100 |

1. Изисквания за документацията

|  |  |  |
| --- | --- | --- |
| № | Изискване | Точки |
| 1 | README.md файл с описание на проекта, инструкции за инсталация и стартиране | 15 |
| 2 | Документиране чрез docstring и коментари в кода | 10 |
| 3 | Приложена е схема на базата данни (диаграма) | 10 |
| 4 | Обяснена е интеграцията между AI модула и уеб приложението | 10 |
| 5 | Изготвени са минимум 10 user stories по Agile методология | 10 |
| 6 | Водени са sprint логове веднъж седмично | 15 |
| 7 | Използвана е система за управление на задачи (напр. Trello, GitHub Projects) със скрийншот на дъската | 10 |
| 8 | Изготвен е краен PDF отчет с описание на постигнатото, резултатите и разпределението на задачите | 10 |
| 9. Правилно използване на контрол на версиите: | Всеки фийчър си има бранч и история на разработване. ПРОЕКТИ БЕЗ РАВНОМЕРНИ И РИТМИЧНИ КОМИТИ НЯМА ДА БЪДАТ ОЦЕНЕНИ. | 10 |
|  | ТОТАЛ | 100 |

I.  Linear Regression  
1.  House Price Predictor — Predict house prices from size, number of rooms, etc. - Стела Георгиева  
2.  Student Score Estimator — Predict exam score based on hours studied. - Мартин Стоянов  
3.  Calories Burned Estimator — Predict calories burned from workout time or heart rate. - Станислав Дженевиз  
4.  Car Price Estimator — Predict used car price from mileage, age, brand, etc. - Лазар Апостолов

II.   Logistic Regression  
5.  Email Spam Classifier — Classify emails as spam or not. - Сергей Топтунов   
6.  Admission Predictor — Predict if a student gets admitted based on GPA and test scores. - Григорий Калашник  
7.  Social Media Ad Click Predictor — Predict if a user clicks on an ad. - Александър Сивков  
9.  Mental Health Survey Classifier - Predict likelihood of depression/anxiety based on survey answers (e.g., PHQ-9 subset). - Зафир Динев

III.   Perceptron  
10.  Binary Image Classifier (e.g. circle vs square). - Димитър Бяндов  
11.  Simple Linearly Separable Classifier — Classify points in 2D space. - Добрил Пеев   
12.  Language Detector - Predict the language of a short text snippet (e.g., English, Spanish, French). - Иво Бадалов

IV.   Neural Network (Basic feedforward)   
14.  Weather-based Outfit Suggestion — Predict what to wear based on temperature and forecast. - Валентин Пендашев  
15.  Rock-Paper-Scissors Classifier — Based on image input or move patterns. - Ивайло Бъчваров

8.  Loan Approval Predictor — Predict if a loan gets approved based on income, age, etc.

13.  Handwritten Digit Recognizer — Use MNIST (start with 2-3 digits only).

**7. Job Application Success – Stanimir Popov**