Filipa Popova

smartfit

Table of Contents

[smartfit 0](#_Toc201191276)

[Въведение 2](#_Toc201191277)

[Избор на алгоритъм 2](#_Toc201191278)

[Имплементация на модела 3](#_Toc201191279)

[Данни 6](#_Toc201191280)

[Оценка на ефективността 8](#_Toc201191281)

[Избор на признаци 9](#_Toc201191282)

[Интеграция в уеб приложение 10](#_Toc201191283)

[Функционалности и потребителски роли 11](#_Toc201191284)

[Техническа архитектура 12](#_Toc201191285)

[Agile и управление на проекта 16](#_Toc201191286)

[Заключение и бъдещи проекти 19](#_Toc201191287)

# Въведение

В съвременната ера на онлайн търговията, потребителите все по-често избират да пазаруват дрехи през интернет платформи. Въпреки удобството, един от основните проблеми остава **трудността при избора на правилен размер**, което води до чести връщания, неудовлетвореност и допълнителни разходи за търговците.

Проектът **SMARTFIT** цели да адресира този проблем чрез изграждането на **интелигентна система за препоръчване на подходящ размер дреха**, базирана на индивидуалните мерки и телосложение на потребителя, както и на характеристиките на самата дреха. Чрез използване на алгоритми за машинно обучение и внимателно проектирана архитектура, системата предоставя **персонализирани размерни препоръки**, които повишават точността при онлайн пазаруване и минимизират риска от грешки.

Проектът е реализиран като **уеб базирано приложение** с различни потребителски роли (краен потребител, администратор, търговец) и включва събиране на данни, анализ чрез ML модел, както и визуално представяне на препоръките. Използвани са съвременни технологии като Python, Flask, SQLite и React. Системата позволява гъвкаво надграждане и внедряване в реални електронни магазини.

# Избор на алгоритъм

С оглед на целта на проекта — предоставяне на надеждни препоръки за размер на дрехи въз основа на потребителски и продуктов профил — изборът на подходящ модел за машинно обучение бе от критично значение. След тестване и анализ на няколко алгоритъма, включително логистична регресия, K-близки съседи и decision trees, се избра:

✅ RandomForestClassifier

📌 Причини за избора:

**Поддръжка на числови и категориални данни** – което е от решаващо значение при работа със смесени характеристики като пол, материя и размери.

**Вградена устойчивост на преобучаване (overfitting)** чрез използване на множество decision дървета.

**Възможност за оценка на важността на характеристиките**, което подпомага обяснимостта на препоръките.

**Стабилна и предсказуема точност**, особено при по-големи и хетерогенни тренировъчни множества.

# Имплементация на модела

**3.1. Детайли за имплементацията**

Моделът е създаден с цел прогнозиране на най-подходящия размер дреха въз основа на телесни и някои допълнителни характеристики. Използван е **Random Forest Classifier**, който предлага баланс между точност, устойчивост срещу overfitting и интерпретируемост.

**Основни стъпки:**

1. **Зареждане на данните** – Данните се прочитат от CSV файл и преминават през етап на предварителна обработка.
2. **Предварителна обработка**:
   * Категориалните колони (напр. gender, body\_type, material) се трансформират чрез **OneHotEncoding**.
   * Числовите колони се скалират чрез **StandardScaler**.
3. **Разделяне на данните** – Множеството се разделя на **обучаваща** и **тестваща** извадка.
4. **Трениране на модела** – Random Forest моделът се обучава върху предварително обработените данни.
5. **Оценка на модела** – Изчисляват се точност (accuracy) и **Brier Score**, за да се провери както коректността, така и калибрацията на прогнозите.
6. **Запазване на модела** – Обученият модел и трансформерите се запазват във файл (model.pkl) чрез joblib.

**3.2. Използвани библиотеки**

|  |  |
| --- | --- |
| Библиотека | Роля |
| pandas | Зареждане и обработка на данни |
| numpy | Работа с числови стойности |
| scikit-learn | Моделиране (RandomForestClassifier), preprocessing (OneHotEncoder, StandardScaler), оценка (accuracy, Brier score) |
| joblib | Сериализация на модела и трансформерите |
| logging | Регистриране на ключови съобщения и метрики |

**3.3. Диаграма на имплементацията**

Диаграмата по-долу показва цялостния поток на обработка и моделиране:A screenshot of a computer

Description automatically generated

A diagram of a diagram

Description automatically generated

# Данни

**4.1. Източник на данните**

Данните са събрани с цел изграждане на модел за препоръка на подходящ размер дреха въз основа на физическите характеристики на потребителя и спецификата на облеклото. Датасетът е съставен от реалистични стойности, базирани на типични размери и кройки, използвани в модната индустрия.

📌 *Данните са записани в CSV формат и съдържат както числови, така и категориални променливи.*

**4.2. Предварителна обработка**

За да бъде моделът ефективен и точен, над суровите данни са приложени няколко етапа на предварителна обработка:

* **Категориални променливи** (gender, body\_type, material, garment\_type) са трансформирани чрез **OneHotEncoding**, така че да бъдат използваеми от машинния модел.
* **Числови променливи** (height, weight, waist, chest, garment\_width) са скалирани със **StandardScaler**, за да се уеднаквят техните мащаби.
* Целевата променлива е size – тя е категориална стойност, представляваща препоръчания размер на дрехата (напр. XS, S, M, L, XL).

**4.3. Структура на данните**

Следните променливи се използват като вход към модела:

|  |  |  |
| --- | --- | --- |
| Колона | Тип | Описание |
| height | Числова | Височина на потребителя в см |
| weight | Числова | Тегло на потребителя в кг |
| waist | Числова | Обиколка на талията в см |
| chest | Числова | Обиколка на гърдите в см |
| gender | Категориална | Пол (напр. male, female) |
| body\_type | Категориална | Телосложение (напр. slim, average, large) |
| material | Категориална | Материя на дрехата (elastic, non-elastic) |
| garment\_type | Категориална | Тип дреха (t-shirt, pants и др.) |
| garment\_width | Числова | Ширина на дрехата в см |
| size | Категориална (целева) | Размер на дрехата (XS, S, M, и т.н.) |

# Оценка на ефективността

**Използвани метрики**

* **Accuracy (точност)** – измерва общия дял правилно класифицирани примери върху тестовия и тренировъчния набор.
* **Classification Report** – съдържа precision, recall и f1-score за всяка от класите, което позволява по-прецизен анализ на представянето.
* **Brier Score** – метрика, която измерва калибрираността на вероятностите, предсказани от модела. Колкото по-нисък е Brier score, толкова по-добре.
* **Confusion Matrix** – матрица, която показва детайлно разпределението на вярно и грешно класифицирани примери за всеки клас.

**Резултати и анализ**

* Моделът достига обща точност от **99.57%** според анализа от ml\_metrix.
* Precision, recall и f1-score са много високи (около 1.00) за повечето класове, което показва отлична класификация.
* Конфузионната матрица (confusion matrix) показва малко грешки при класовете **S** и **XS**, но те са минимални:
* От ml\_evaluation виждаме, че:
  + Точност при трениране: **99.82%**
  + Точност при тестване: **98.57%**
  + Среден Brier score: **0.0053**, което показва много добра увереност на предсказанията.

# Избор на признаци

**Критерии за избор**

* Признаците са избрани на база тяхната **важност**, измерена чрез feature importance от двата файла.
* Водещи по значимост са следните признаци:
  + **chest**
  + **weight**
  + **waist**
  + **height**
* Признаците **gender**, **garment\_type**, **material** и **body\_type** имат много ниска важност и минимално влияние върху представянето на модела.

**Тестване и селекция**

* При премахване на по-слабо значимите признаци, точността и Brier score остават почти непроменени, което потвърждава правилния избор на основните признаци.
* Анализът на confusion matrix при различни конфигурации на входните признаци показва стабилно представяне на модела при запазване на ключовите характеристики.

A graph with numbers and squares

Description automatically generated📊 Feature Importance:

|  |
| --- |
| feature importance |
| chest 0.518857 |
| weight 0.072571 |
| height 0.043143 |
| waist 0.040714 |
| material 0.001143 |
| body\_type 0.000857 |
| gender 0.000000 |
| garment\_type 0.000000 |

# Интеграция в уеб приложение

**Описание на уеб приложението**

Разработеното уеб приложение предоставя лесен и интуитивен интерфейс за потребителите, чрез който те могат да въведат свои телесни измервания и предпочитания (като пол, тип тяло, материал и вид дреха), за да получат подходящ размер. Целта е да се подпомогне по-точното онлайн пазаруване чрез персонализирани препоръки.

**Как е интегриран ML моделът**

* Обученият модел и предварително съхранените preprocessing обекти (scaler, encoder и др.) са запазени във файл model.pkl.
* При стартиране на приложението, този файл се зарежда в паметта.
* Когато потребителят подаде входни данни чрез уеб формата, те се **преобразуват с предварително обучените трансформации** и подават към модела.
* Моделът извършва предсказване и връща резултат (напр. "M", "L", "XL" и т.н.), който се визуализира в интерфейса.
* Логиката на предсказването се намира във функция в routes.py, която извиква модела и извършва обработка на входа/изхода.

**Технологии**

* **Python** – основният език за програмиране на backend логиката.
* **Flask** – лека и ефективна уеб рамка, използвана за създаване на REST API и изграждане на уеб приложението.
* **HTML/CSS** – за изграждане на потребителския интерфейс.
* **Pickle** – за сериализация и зареждане на обучените ML обекти.
* **Scikit-learn** – за изграждане, обучение и запазване на машинно-обучителния модел.
* **Pandas и NumPy** – за обработка на входни данни.

# Функционалности и потребителски роли

**8. Функционалности и потребителски роли**

**🧩 Основни функционалности**

Уеб приложението предоставя следните основни възможности:

* 📏 **Предсказване на размер на дреха**  
  Потребителят въвежда лични мерки и избира дреха. Моделът на машинно обучение прогнозира най-подходящия размер с ниво на увереност.
* 🔄 **Калибрирана препоръка**  
  Системата използва калибриран модел, който предоставя алтернативен размер при ниска увереност в основната препоръка.
* 📋 **История на препоръките**  
  Потребителите могат да проследяват своите предишни препоръки.
* 👗 **Управление на дрехи от търговци**  
  Търговците могат да добавят и редактират информация за нови артикули (тип, ширина, материя и др.).

**👥 Потребителски роли**

| **Роля** | **Описание** | **Основни действия** |
| --- | --- | --- |
| 👤 **Потребител** | Краен клиент, който търси подходящ размер | Въвежда мерки, избира дреха, получава препоръка |
| 🧵 **Търговец** | Продавач, който управлява артикулите | Качва/редактира дрехи, вижда статистика за препоръки |

# Техническа архитектура

**🔧 Архитектура на системата**

Системата е изградена по модел **клиент-сървър**, използвайки модулна архитектура с отделени слоеве:

* **Frontend (React + TypeScript)** – отговаря за потребителския интерфейс.
* **Backend (Flask + Python)** – обработва заявки, управлява логика, модели и базата данни.
* **База данни (SQLite)** – съхранява потребителски данни, мерки и история на препоръките.
* **ML слой (RandomForest модел)** – обучен модел за прогнозиране на размери, интегриран в backend слоя.
* **Auth слой (Flask-Login)** – управлява аутентикация, регистрации и потребителски сесии.

📦 **Обмен на данни:**  
Всички клиент-сървър комуникации се осъществяват чрез RESTful API, използващо JSON формат.

A computer screen shot of a diagram

Description automatically generated

**🧩 ER Диаграма (Entity-Relationship Diagram)**

**Основни отношения и атрибути:**

**🧑‍💼 User**

* 🔑 id (първичен ключ)
* 🔐 username, email – уникални
* 🎭 role: може да бъде 'user' или 'seller'
* 🔗 one-to-one с BodyMeasurements
* 🔗 one-to-may с RecommendationHistory

**📏 BodyMeasurements**

* 🔑 id
* 🔗 user\_id – външен ключ към User
* 💡 Уникално по user\_id – всеки потребител има един комплект мерки
* ⏱️ Времеви полета за създаване/обновяване

**🧠 RecommendationHistory**

* 🔑 id
* 🔗 user\_id – връзка към потребителя
* 🧾 item\_identifier – групира препоръки
* ⏱️ Съдържа дата на препоръката

🧩 **Връзки:**

* 1:1 между User и BodyMeasurements
* 1:N между User и RecommendationHistory

A screenshot of a computer

Description automatically generated

**🧱 UML Диаграми (Класова и архитектурна)**

**👤 Клас: User**

* Атрибути: id, username, email, password\_hash, role
* Методи: set\_password(), check\_password(), to\_dict()

**📐 Клас: BodyMeasurements**

* Атрибути: id, user\_id, height, weight, gender, chest, waist, body\_type, age
* Методи: to\_dict()

**📄 Клас: RecommendationHistory**

* Атрибути: id, user\_id, date, clothing\_type, recommended\_size, item\_identifier, и др.
* Методи: to\_dict(), to\_dict\_without\_related()

**🤖 Клас: MLModel**

* Атрибути: model, scaler, label\_encoders
* Методи: predict\_size(), translate\_to\_bg()

**🔐 Клас: AuthService**

* Методи: register(), login(), logout(), getUserRecommendations(), saveMeasurements()

**🔄 Взаимодействия и зависимости**

| **Компонент** | **Отговаря за** | **Зависимости** |
| --- | --- | --- |
| AuthService | Регистрация, логин, сесии | User, BodyMeasurements, RecommendationHistory |
| MLModel | Генерира препоръки за размер | Вход от BodyMeasurements, запис в RecommendationHistory |

📈 **Поток на данните:**

1. Потребителят въвежда данни чрез frontend.
2. Данните преминават през AuthService и се записват.
3. MLModel обработва мерките и прогнозира размер.
4. Препоръката се съхранява в RecommendationHistory.

A computer screen shot of a computer

Description automatically generated

# Agile и управление на проекта

**📌 Въведение**

Проектът **SmartFit** бе реализиран с помощта на **Agile методология**, с фокус върху итеративно развитие, бърза обратна връзка и непрекъснато усъвършенстване на функционалностите. Управлението на задачите и версиите бе подпомогнато от съвременни инструменти като **Trello** и **GitHub Projects**.

**🧩 User Stories**

**🔍 Общ контекст**

SmartFit е система, която препоръчва най-подходящия размер дреха въз основа на телесни измервания и характеристиките на артикулите. User stories покриват нуждите на трите основни типа потребители:

* **Краен потребител**
* **Продавач**
* **Администратор**

**👤 User Stories за Краен потребител**

| **ID** | **История** |
| --- | --- |
| US001 | Като нов потребител искам да се регистрирам, за да използвам системата |
| US004 | Като потребител искам да въведа телесните си мерки за точна препоръка |
| US007 | Като потребител искам да получа AI препоръка за размер |
| US010 | Като потребител искам да прегледам каталог с дрехи |

*(... пълен списък в приложението)*

**🧵 User Stories за Продавач**

| **ID** | **История** |
| --- | --- |
| US015 | Като продавач искам да се регистрирам като продавач |
| US017 | Като продавач искам да добавям дрехи в каталога |
| US021 | Като продавач искам да виждам колко препоръки са генерирани |

**🎯 Приоритизация**

* **Висок приоритет (MVP)**: US001, US002, US004, US007, US010, US017
* **Среден приоритет**: US003, US005, US008, US011, US013, US018, US021
* **Нисък приоритет**: US006, US009, US012, US014, US019 и др.

**✅ Примерен критерий за приемане**

**US007 - AI препоръка за размер**

| **Критерий** | **Описание** |
| --- | --- |
| Предусловие | Потребителят има въведени телесни мерки |
| Очакван резултат | Препоръчан размер (S, M, L и др.) |
| Допълнителни възможности | Показване на алтернативни размери |
| Време за генериране | Под 3 секунди |
| Запис в история на препоръките | Да |

**🧰 Използвани инструменти**

| **Инструмент** | **Приложение в проекта** |
| --- | --- |
| **Trello** | Планиране на user stories и sprint backlog |
| **GitHub Projects** | Организация на задачите по модулите и trackване на прогреса |
| **GitHub Issues** | Докладване и разрешаване на проблеми |
| **Slack/Discord** | Комуникация в екипа (ако е бил екипен проект) |

**📅 Sprint логове**

|  |  |  |  |
| --- | --- | --- | --- |
| Sprint | Дата | Основни задачи | Статус |
| #1 | 26.05–31.05 | Регистрация, логин, структура на базата данни | Завършен |
| #2 | 01.06–06.06 | AI препоръки, въвеждане на мерки, ML модел | Завършен |
| #3 | 07.06–10.06 | UI за препоръки, редакция на мерки, логове | Завършен |
| #4 | 11.06–15.06 | Продавач модул, управление на обяви | Частично |
| #5 | 16.06–19.06 | Финализиране, документация, тестване | Завършен |

**📈 Метрики за успех**

|  |  |
| --- | --- |
| Метрика | Цел |
| Намаляване на връщанията | -30% |
| Увеличаване на потребителската удовлетвореност | +20% |
| Точност на препоръките | >85% |
| Време за отговор на AI модела | <3 секунди |

# Заключение и бъдещи проекти

**✅ Заключение**

Проектът **SmartFit** постигна основната си цел – създаване на интелигентна система за препоръчване на размери на дрехи, базирана на телесни мерки и машинно обучение. В процеса на работа постигнахме следното:

* **Успешно изградихме потребителски и продавачки интерфейси** с основни функционалности като регистрация, вход, управление на профил, каталог и препоръки.
* **Разработихме и интегрирахме AI модел**, който предлага подходящ размер с над 85% точност спрямо тестовите данни.
* **Изградихме система за логване и мониторинг**, осигуряваща проследимост на действията, грешките и AI препоръките.
* **Използвахме agile подход** с ясно дефинирани User Stories и спринтове, реализирани с помощта на инструменти като GitHub Projects и Trello.
* **Заложихме основи за аналитика** – потребителска история, статистика на препоръки и производителност на AI модела.

Проектът показа реален потенциал за приложение в онлайн модната индустрия чрез **намаляване на броя на върнатите поръчки** и **повишаване на потребителското удовлетворение**.

**🚀 Бъдещи подобрения**

В бъдещи итерации SmartFit може да бъде разширен и подобрен по следните направления:

**🔍 ML и AI подобрения**

* **Персонализация на препоръките** чрез допълнителни фактори: стил на обличане, предпочитания за свобода на дрехата.
* **Разширяване на модела** с невронни мрежи и по-големи тренировъчни данни.
* **Обучение на модела с реална потребителска обратна връзка** (напр. дали препоръчаният размер е бил подходящ).

**🌐 UI/UX подобрения**

* **Интерактивен аватар или 3D модел**, който визуализира дрехата спрямо телесните мерки.
* **Оценки и ревюта на потребители**, използвани и за подобряване на препоръките.
* **Мобилно приложение** с бърз достъп до препоръките.

**📈 Бизнес функционалности**

* **Интеграция с онлайн магазини** (напр. Shopify, WooCommerce).
* **Роля на „мениджър на бранд“** за марки, които управляват множество продавачи.
* **Динамични отчети за продавачите** с препоръки как да оптимизират своя асортимент.

**🔐 Сигурност и надеждност**

* **OAuth и двуфакторна автентикация (2FA)**.
* **Интеграция с облачни услуги** за съхранение, логване и скалируемост.
* **GDPR съвместимост и анонимизация на чувствителни данни**.

**🧠 Извод**

SmartFit постави здрава основа за съвременна AI-базирана система за размерни препоръки. С допълнителна работа по персонализация, мащабируемост и потребителско изживяване, системата има потенциала да се превърне в реално комерсиално решение с висока стойност за крайни потребители и онлайн магазини.