CLIENT REPORT

TITLE: SMARTSIZE AI: MACHINE LEARNING-BASED CLOTHING SIZE RECOMMENDER

CLIENT: FASHION RETAILERS, E-COMMERCE BUSINESSES & CONSUMERS WITH SPECIFIC TASTES

PROJECT TEAM:

HARSHINI S. R – 2420030113

R. L. S GIRIDHAR – 2420030171

SATYADEV VETA – 2420090120

1. EXECUTIVE SUMMARY

Introduction: The Sizing Conundrum in Apparel E-Commerce

The high incidence of product returns that is typical of digital apparel creates a burden on the sector, which directly relates to lower customer satisfaction and huge logistics costs. Inherent vagueness and heterogeneity of traditional garment sizing charts stand as a primary antecedent of this problem. These static guides vary significantly among brands, across geographic regions, and for different product categories, such as athletic wear versus formal wear. This inconsistency leads to suboptimal consumer choice, purchase errors, and a general erosion of consumer confidence.

Solution: SmartSize AI Platform

SmartSize AI is an intelligent, data-driven platform conceived to offer a definitive solution to this problem faced across the industry. The advanced garment size recommendation engine ideally reduces return rates, enhances customer satisfaction, and equips apparel businesses with the powerful tool necessary for very personalized size suggestions.

Core Methodology and Technological Framework

The platform abandons the limitations of static charts in favor of a dynamic, analytical framework. SmartSize AI uses an advanced machine learning architecture to present the most precise fit recommendations in real-time.
The system works by processing and correlating two main multidimensional data streams:

- User-specific anthropometric data: Volumetric and linear measurements, either provided directly by the user or inferred from historical data.
- Garment-specific technical specifications: granular data that encompasses not only standard measures, but also fabric composition, material elasticity, design cut, and manufacturing tolerances.
- Such complex datasets integrated into the platform's predictive algorithms calculate the ideal fit for any particular user-item pairing and present it synchronously at the time of the digital shopping journey.

Value Proposition and Operational Impact

The combination of SmartSize AI's machine learning models with an intuitive, low-friction user interface yields quantifiable operational and commercial benefits. Retailer: The main value proposition is an improvement witnessed in the fit accuracy, due to which the root cause of the returns was identified. In addition, the aggregated data presents actionable analytics that could be used to optimize inventory planning, refine supply chain logistics, and help with future product development. To the Consumer, the technology provides seamless, reliable, and confidence-inspiring shopping. SmartSize AI removes cognitive load from the consumer by translating complex garment data into a simple, accurate recommendation, thereby enhancing the probability of conversion and fostering long-term brand loyalty.

2. CLIENT PROBLEM / BUSINESS NEED

Retailers continue to deal with the ongoing issues of sizing inconsistencies:

- High rates of returns from customers that cite poor fit
- Customer dissatisfaction stemming from inaccurate size charts
- Loss of trust in the brands due to previous inconsistencies with sizing
- Increased logistics and restocking costs, and the cost of reverse shipping

Consumers are more likely to simply guess or engage in trial-and-error when it comes to selecting the size they want, especially for items purchased in online marketplaces where consumers don't have fitting rooms. Providing a solution that leads to better size accuracy will benefit both the customers and the retailers.

3. CLIENT MEETINGS

Client Meeting 1: Project Understanding & Data Preparation

1. What is the main goal of the SmartSize AI project? Answer: The main objective is to conceptualize, design, and deploy an AI-powered system engineered to act as an intelligent size recommendation engine. It will predict the ideal clothing size of a diverse user base by processing a set of core user-provided attributes such as height, weight, gender, age, and subjective fit preference, for example, tight, regular, loose. The ultimate aim is to fundamentally enhance the online shopping experience by replacing ambiguous static size charts with a data-driven, personalized, accurate recommendation.

2. Why is predicting the right size of clothes important? Answer: Accurate size prediction is important for two reasons: it responds to critical challenges in e-commerce.

Logistical & Financial: Size mismatches are the single largest driver of product returns in the online apparel industry. These returns incur substantial reverse logistics costs of shipping, restocking, and processing; more often than not, the inventory must be sold at a discount.

Customer Experience: Poor fits automatically cause customer dissatisfaction, deteriorate brand trust, and add friction to the shopping process. By contrast, a good prediction gives buyer confidence, reduces anxiety during the buying process, and substantially improves the chances of repeat business and brand loyalty.

3. How does SmartSize AI improve the user experience? Answer: SmartSize AI revolutionizes the user experience by eliminating the primary points of friction that exist in online apparel purchasing. Instead of making users decipher confusing, non-standardized size charts or "guess" their size based on previous, often incorrect, purchases, the platform provides an immediate, data-driven, and personalized recommendation. This removes the cognitive load and uncertainty from the user, streamlining the path from product discovery to purchase confirmation while minimizing the post-purchase dissonance associated with "fit anxiety."

4. What problem does this project solve in the fashion industry? Answer: This project squarely addresses the systematic problem of unstandardized sizing, which has been the bane of the fashion industry for decades and has become even more acute with the shift to e-commerce. There is no such thing as a universal "Medium." Sizing varies dramatically across brands, geographic markets, and even different items within the same brand. SmartSize AI acts as a universal translator, abstracting that complexity away from the customer and making data-driven purchasing decisions possible in a landscape of high inconsistency.

5. Why was machine learning chosen? Answer: For this problem, machine learning was the only viable technological choice; one cannot envision any other practical implementation of technology. A static, rule-based system-for example, "if height > 6ft and weight > 200lbs, then size = XL"-is decidedly too rigid to model the highly non-linear relationships between human anthropometrics and garment sizing. Machine learning allows such a system, through a supervised

classification model, to learn the intricate patterns from a vast dataset to generalize predictions for new, unseen user combinations with an accuracy and personalized fit that a static system could not possibly achieve.

6. Which dataset was used to train the model? Answer: The idea is to create an algorithmically generated synthetic dataset to bootstrap the project and also to validate the model architecture without compromising user privacy. The synthetically generated dataset includes key user attributes: Height (in cm), Weight (in kg), Gender (Male, Female), Fit Preference (Tight, Regular, Loose), and Age. Each of these features was programmatically mapped to one of four size labels: S, M, L, or XL. This synthetic approach provided a clean, balanced, and complete dataset for initial training and prototyping.

7. How were features selected? Answer: Feature selection was guided by domain expertise and an analysis of existing public datasets, including data from other fashion e-commerce companies. The selected features Height, Weight, Gender, Age, and Fit Preference are the most important and generally available points that have a direct consequence on body dimensions and, therefore, on clothing fit. 'Fit Preference' was especially considered crucial because it adds a layer of personalization; two users can have identical bodies yet may want different sizes because of their fit preferences.

8. Why were categorical features like gender and fit preference encoded? Answer: Fundamentally, all machine learning models are mathematical and work with numbers. They cannot innately process string-based categorical labels such as "Male" or "Loose." Thus, these features needed to be encoded, meaning transformed into a numerical representation. In this project, Label Encoding was adopted, where each category is mapped to a unique integer; for example, S=0, M=1, L=2, XL=3. This transformation is a mandatory preprocessing step that makes the data "intelligible" to the model's algorithm.

9. How was missing data dealt with? Answer: One great thing about using a synthetically generated dataset for this prototype is that there were no missing values; the data were created to be complete. However, for real-world deployment, the pre-processing pipeline has been architected to provide handling for potential missing data NaN values, whereby missing data would be imputed with a default value, such that the application will not crash if, for some reason, a user does not provide an input for a given field.

10. What preprocessing was applied to the data? Answer: A two-stage preprocessing pipeline was applied:
Label Encoding: As explained above, all categorical features, such as Gender and Fit Preference, and also the target variable Size, were converted to integers from text.
Standard Scaling: The numerical features, which are Height, Weight, and Age, have been normalized using StandardScaler. It rescales the data to have an approximate mean of 0 and a standard deviation of 1. This is important because features with different scales-for example, Weight ranges between 50-150 and Age ranges between 18-70-might disproportionately inform the model. Scaling makes sure all features contribute equally in the prediction, which is essential for many algorithms, like KNN and Logistic Regression, and it enhances convergence and improves the performance of all models.

Client Meeting 2: Model Development & Evaluation

11. Which machine learning models have been tested? Answer: A comparison was done by training and evaluating four different types of supervised classification models. The candidates were:

Logistic Regression: Baseline linear model that establishes a performance floor.

K-Nearest Neighbors: A distance-based model on how "closeness" in the feature space relates to size.

Decision Trees: An interpretable, tree-based model.

Random Forest: This is an ensemble model (multiple decision trees) that is highly accurate and robust.

12. Why was the Decision Tree chosen as the final model? Answer: While the Random Forest showed marginally higher accuracy, the Decision Tree was selected as the final production model for two critical strategic reasons:

Interpretability: Decision Trees are a "white-box" model. We can actually directly visualize the decision-making logic behind the models- for instance, "IF Weight > 75kg AND Height < 170cm THEN.". This is especially of value in debugging, demonstrations to the client, and understanding why a model makes a particular recommendation.

Performance and Efficiency: It provided a strong balance of high accuracy, over 90%, with fast training time and quick predictions, latency-which worked best for a real-time web application. Also, inherently handles both categorical and continuous variables very effectively.

13. How were the hyperparameters tuned? Answer: A "naive" Decision Tree can easily overfit the data. In order to avoid this and improve its performance, some of the most important hyperparameters were tuned using Grid Search along with Cross-Validation. This automated process tested various combinations of parameters, including:

max_depth: the maximum "depth" of the tree, to control its complexity.

min_samples_split: The minimum number of samples required to split a node.

Criterion: A function to measure the quality of a split. Both "gini" and "entropy" were tried. It was an important optimization that helped to improve the accuracy on unseen data and to make the model generalize well.

14. What performance metrics were used? Answer: A comprehensive evaluation suite was used, as "accuracy" alone can be misleading:

Accuracy Score: The main metric, which is the proportion of accurately predicted values.

Confusion Matrix: An image table listing performances for each class, specifying where the model went wrong; for example, 'M' predicted as 'L'.

Classification Report: This gave the most granular detail, providing the performance broken down by class using:

Precision: Out of all the times it predicted 'L', how many were correct? (Minimizes false positives).

Recall: Of all the real 'L' sizes, how many did it correctly identify? (Minimizes false negatives).

F1-Score: The harmonic mean of Precision and Recall, providing a single score that balances both.

15. How was model validation performed? A rigorous validation strategy was adopted. First, the entire dataset was partitioned by using a standard 80:20 train-test split. The model was trained exclusively on the 80% training set and then evaluated on the 20% "unseen" test set to get an unbiased estimate of its performance. Furthermore, k-fold cross-validation with k=5 was used during the hyperparameter tuning phase. It means the training data is divided into 5 "folds," training on 4 and testing on 1, then rotating—this ensures the performance of the model is stable and does not depend on one particular random split.

16. What were some of the challenges during training? Answer: The major challenges faced were:

Initial Overfitting: Initial Decision Tree models had an overall training accuracy of 100%, coupled with poor test accuracy - a clear sign of "memorizing" the data. The solution was tuning max_depth.

Inconsistencies with Feature Scaling: Models such as KNN and Logistic Regression performed very poorly until scaling was applied because the 'Weight' feature was overpowering the 'Age' feature.

Class Imbalance (Minor): The initial generation of synthetic data was slightly unbalanced in terms of classes. This was rectified through resampling techniques so that the model got an equal example count for each size to prevent bias toward one particular majority class.

17. How does the model make predictions for data it has not seen? Answer: The model generalizes to new, unseen data by applying logical rules learned from its training. When a new user inputs their data (e.g., 180cm, 80kg, Male, 30yrs, Regular fit), the model "drops" this data point through its tree structure. It follows learned branches—a series of "if-then-

else" questions based on the input features—down to a final "leaf node," which contains the predicted size (e.g., 'L'). This generalization capability was validated with a high accuracy score on the test set.

18. Why was overfitting a concern? Answer: Overfitting is the main problem of Decision Trees, which, if not constrained, will go on splitting and multiplying until each data point in the training set is perfectly classified. The result is a very deep tree that "memorizes" the noise and outliers in the training data. When exposed to new data, this "memorized" model performs terribly, failing to generalize. Limiting the depth and pruning the tree are the main techniques of combating this.

19. How was model accuracy improved? Answer: Accuracy was systematically improved through an iterative process.

Better Preprocessing: Using StandardScaler for feature normalization.

Hyperparameter tuning: the use of Grid Search to obtain an optimal max_depth and min_samples_split that balanced simplicity against predictive power.

Feature Engineering: Not very necessary in this case, but experiments were run to see if the combination of features would help; in this case, raw height/weight was better.

Data Balancing: Ensuring the model was trained on an equitable distribution of all size classes.

20. What was the final model accuracy? Answer: After preprocessing, tuning, and validation, the final Decision Tree model yielded a stable and robust accuracy of about 90–92% on the unseen test data. This reflects a high degree of generalization by the model concerning the patterns it has learned, and that this model is quite suitable for production deployment in the client-facing application.

Client Meeting 3: Streamlit Frontend & App Integration

21. What is Streamlit, and why was it used? Answer: Streamlit is an open-source Python framework originally designed for data scientists and ML engineers. It lets developers quickly create an incredibly interactive, shareable web application of data scripts and machine learning models using nothing but Python. It was chosen for this project because it negates the need for traditional, complicated web development such as JavaScript, HTML/CSS, and backend frameworks like Flask/Django. That enabled us to build and deploy a fully functional, professional-looking prototype in a fraction of the time.

22. How does the application collect user input? The application collects user input via a clean, intuitive user interface composed of Streamlit's built-in interactive widgets. These are organized in a sidebar via st. Sidebar to keep the main app area clean.

St.number_input is for numeric features: Height, Weight, Age. Besides, it needs to define the min/max value and step.

Use st. Radio or st. selectbox for categorical features - Gender, Fit Preference. This allows you to specify a pre-defined set of options to a user, avoiding input errors.

23. What happens when the user clicks “Predict”? Answer: When the user clicks the "Predict" button, the call to the st. button triggers an exact sequence of events:

Data Ingestion: The app captures the current values from all of the Streamlit widgets.

Preprocessing: Feed raw input data through the same preprocessing pipeline as that used to train the model; that is, Label Encoding and Standard Scaling. It's a very important step to take so that the data is in the format the model will require.

Conversion: The processed inputs are converted into a 2D NumPy array.

Prediction: The array passed to the .predict() method of a loaded model

Predicted size (for example, 'L') is returned by the model, captured, and displayed back to the user on the screen.

24. How was the model integrated into Streamlit? Answer: The integration is a straightforward, two-part process:

Serialization (Offline): Once the model was trained and finalized in our development environment - say, a Jupyter Notebook - it was "saved" to disk as a single file. We used the joblib library, which is more efficient for ML models containing large NumPy arrays, to serialize the trained model object into a .pkl (pickle) file.

Deserialization (In-App): In the Streamlit application script, app.py loads this .pkl file into memory once - that is, when the app starts for the first time. This deserialized object is the fully trained, ready-to-use model, which the app can then call for live predictions.

25. What are some of the frontend elements that enhance usability? Answer: Other than the core widgets, several UI/UX elements were implemented to enhance the usability and professionalism of the application:

Layout: A clean main-area/sidebar layout to separate the inputs from the results.

Theming: Custom color themes - via config. toml - to align with a potential brand identity.

Visuals: st. An image was used to show a static size_chart.png supplementary reference for the user, building trust and providing context.

Responsive Design: Streamlit is intrinsically responsive, which means the app automatically looks great and works well both on desktop and mobile.

26. Why was there an error with the image file? Answer: A first FileNotFoundError was experienced during development. This was a simple, but common pathing problem. The Streamlit script (app.py) was attempting to load size_chart.png using a relative path, but the file was located in the wrong directory. The solution was to locate the image file in the same root folder as the Python script so that the relative path (st.image("size_chart.png")) would be valid.

27. How are prediction errors handled? Answer: To create a robust application that will not crash, try-except code blocks were set up around the prediction logic. This is an important error-handling practice. If, for some reason, the user's input causes a problem in the preprocessing or prediction phase of the program, such as an unexpected None value, the except code block will intercept the resulting runtime error. Instead of crashing, the app will present a friendly user message-for example, st.error("An error occurred. Please check your inputs.")-and permit the user to try again.

28. What changes were added to the application interface? Answer: The interface is iteratively polished based on feedback:

Input Validation: Added explicit checks and warning messages if a user enters nonsensical data (e.g., Weight = 0).

Spacing & Readability: Employed appropriate usage of st.markdown() and st.header() to create logical sections, better spacing, and clear user instructions.

Presentation of Results: Moved the final prediction to a prominent position, using st.success() with a custom-formatted string such as f"We recommend Size: {prediction}" to make the result clear and celebratory.

29. How does the app show the result? Answer: The final predicted size is displayed using a call to the st.success() function. This visually renders the text output in a colored - by default green - and formatted box. This immediately and positively conveys the "answer" to the user, setting the result apart from the rest of the text and input fields in the app, as well as signaling a successful end

30. Why is Streamlit ideal for prototypes? Answer: Streamlit is the ideal tool for projects at this stage (academic, demo, or prototype) for three major reasons:

Speed: It enables development from a model script to a live web app in hours, not weeks.

Interaction: It allows for real-time interaction, whereby clients and stakeholders can "play" with the model and understand immediately its value.

Simplicity: It's "just Python." This removes the barrier to entry for data scientists who aren't full-stack developers and simplifies the entire deployment and iteration cycle.

Client Meeting 4: Deployment, Ethics & Future Enhancements

31. What are some deployment methods? Answer: The application is ready to be deployed publicly. There is a direct path with Streamlit Cloud, which is free and links to the GitHub repository of the project, automatically handling all server configurations and dependencies. For a more scalable, enterprise-level deployment, one could containerize the Streamlit application (via Docker) and deploy it to hosting services like Heroku, AWS (EC2/ECS), or Google Cloud Run, which will give more precise control over resources and infrastructure.

32. How will model updates be managed? Answer: The architecture is designed to do easy "CI/CD" (Continuous Integration/Continuous Deployment) of the model. For example, when a new, improved model is trained, model_v2.pkl, the update can be done simply by:

Replace the old .pkl file in the GitHub repository.

Committing the change. Streamlit Cloud will automatically detect the change, "re-build" the app, and the new model will be live within minutes with zero downtime and no changes required to the frontend code.

33. Why is user feedback important? Answer: User feedback is the most critical component for moving beyond the synthetic dataset. A feedback mechanism -e.g., "Was this recommendation helpful? Yes/No"-is the primary way to collect real-world, ground-truth data. The feedback loop is useful in the following ways:

Edge Cases: Circumstances in which the model is constantly wrong.

Data Drift: Changes in user behavior or product sizing over time.

New Features: User requests for new features, such as brand-specific recommendations.

34. How can the dataset be expanded? Answer: The synthetic dataset was a starting point. The following is the road map for expansion in the future:

Collecting Real User Data: Anonymously collecting the feedback data - user inputs + "Did it fit?".

Granularity Addition: Newer, more precise features to be added are body measurements, if the users agree to provide them: chest, waist, and hip.

Brand-Specific Data: Integrating brand-specific size charts to create multiple models that represent each brand's unique sizing, rather than creating a single general model.

35. What are the scalability plans? Answer: The current Streamlit app is sufficient for demo purposes, but it is not designed for high-traffic e-commerce. The scalability plan involves a microservice architecture: API Creation: The core model, along with preprocessing logic, will be wrapped in a REST API using any preferred framework such as FastAPI or Flask. Decoupling: This API will be deployed as a scalable "prediction service" independently, such as on AWS Lambda. Integration: This API would be invoked by the main website of the e-commerce platform to get a prediction so as to handle thousands of concurrent user queries.

36. Why is data privacy important? Answer: Data privacy is a non-negotiable ethical and legal requirement. User attributes such as height, weight, gender, and age are elements of Personal Identifiable Information or at least sensitive personal data. The system shall be designed in conformance with regulations, such as GDPR or CCPA, where all data collected shall be anonymized, securely stored, and used only for its declared purpose. Only then will the project have any chance of success because transparency in privacy policy builds users' trust in it.

37. How can deep learning improve the results? Answer: While the Decision Tree is effective, Deep Learning could offer a leap in precision once the dataset becomes more complex. A neural network might learn very fine, complex, and non-linear patterns between a much wider array of inputs-like body measurements, fabric type, garment cut, and user attributes-that more simpler models cannot grasp, hence increasing accuracy even further.

38. Which features can be implemented in the future? Answer: The "Future Roadmap" includes several high-value features: Brand-based models: The ability to select a brand (e.g., "Nike", "Levi's") and get a prediction based on that brand's specific sizing. Fabric-based adjustments include a toggle for "Stretchy" vs. "Rigid" fabric, which adjusts the recommendation. User History: A "MyFit" profile, where users can save their measurements and/or track their past purchases to have the model personalize to their history. Image-Based Sizing: A more advanced feature where customers can upload a photo for body-shape analysis.

39. How does personalization help in providing the best user experience? Answer: True personalization is much more than a single "fit preference." The ultimate goal is to generate models that adapt to a user's unique preferences for different categories. A user might prefer a "Tight" fit for t-shirts but a "Regular" fit for sweaters. A personalized model would learn these subtle characteristics from their feedback and purchase history, providing personalized recommendations that seem "curated" for them, greatly enhancing user loyalty.

40. Why is continuous learning essential? Answer: The fashion world is not static. Sizing standards change, new brands emerge, and user preferences-e.g., a trend for "oversized" fits-change. The performance of a "set it and forget it" model will inevitably decay. Such a model requires a continuous learning pipeline that automatically retrains the model on new user feedback data at regular intervals-e.g., monthly-so the system stays relevant and accurate, adapting to the dynamic nature of the fashion industry.

4. PROPOSED SOLUTION

SmartSize AI combines:

- Body Measure Inputs (height,   chest, waist, hips, etc.) 1).
- Machine Learning-Based Size Prediction
- Interactive Web Interface (Streamlit)

The platform accepts user-provided measurements and accurately predicts an optimal sizing for clothing. A confidence percentage is also presented, so customers can see how strongly Digit recommends this size, and retailers can intelligently update size charts or product descriptions.

5. VIDEO INTERVIEW

Drive link: Video interview

6. GEOTAG PHOTOS

S
W
E
N
GCW3+WM6, ALEAP Industrial Area, Gajularamaram, Hyderabad, Telangana 500043, India
07 November 2025 15:45:55
Lat: 17.547264, Long: 78.403839
Google

W
S
N
E
GCW3+WM6, ALEAP Industrial Area, Gajularamaram, Hyderabad, Telangana 500043, India
07 November 2025 15:50:00
Lat: 17.547264, Long: 78.403839
Google