**CSCE 5215 - Machine Learning**

**Final Graduate Project (Final Exam)**

**Venkata Sri Satya Sai Kiran Ayyagari – 11713213**

**Introduction:** The demand for precise image classification systems in fashion and e-commerce retailers along with retail operations drives researchers to investigate ensemble learning methods. Fashion MNIST provides 60,000 training samples with 10,000 testing images of clothing items to assess model robustness as a popular benchmark for evaluation. The presented project implements three separate classification methods (SVM, Random Forest and CNN) which cooperate within an ensemble structure to outperform individual models.

**Data Preprocessing:**

**Dataset Loading:** Users could access Fashion MNIST data through the tensorflow.keras.datasets API which provides a standardized reliable method to retrieve image information.

**Data Normalization:** The 28x28 grayscale images use pixel values spreading from 0 through 255. To achieve normalization the images received pixel value division by 255.0 to transform their scope into the [0, 1] range which helps training converge faster.

**Data Reshaping:** SVM alongside Random Forest required all images to be transformed into vectors that contained 784 dimensions. The CNN needed (28, 28, 1) format for representing height, width and channel dimensions.

**Data Augmentation:** The reduction of overfitting in the CNN was achieved through the utilization of ImageDataGenerator with random rotation and shifting and zoom transformations.

**DataSplitting:** The dataset underwent partitioning using train\_test\_split from sklearn.model\_selection while keeping the class distributions intact through stratified sampling methods. It allocated 70% of data for training and 30% for testing purposes.

**Model Development:**

**Support Vector Machine:** The training of the linear SVM happened through utilization of the one-vs-rest methodology. PCA emerged as an optional technique to perform dimension reduction on features for maintaining the maximum information retention.

**Random Forest:** The Random Forest classifier contained 100 trees while employing Gini impurity (criterion=n\_estimators=100) for its decision making. This algorithm reached good performance regarding its fast-processing times and possibility to easily interpret results yet failed to recognize minor spatial dependencies at pixel resolution.

**Convolutional Neural Network:**

* CNN architecture:
* Input Layer: 28x28x1
* Convolution Layers: Two convolutional layers (32 and 64 filters)
* Pooling: MaxPooling2D layers
* Dropout: To reduce overfitting (rate=0.25)
* Fully Connected Layers: Flatten + Dense + SoftMax

The training phase utilized Adam optimizer alongside categorical cross-entropy loss function as its optimization approach. The model demonstrated high testing accuracy because it successfully learned spatial hierarchy structures.

**Ensemble Model:** A soft voting approach was used to merge predictions obtained from SVM and Random Forest and CNN. The prediction models calculated probabilities and used the class with the maximum average probability for their final predictions.

**Model Evaluation:**

**Metrics Used:**

* These metrics present a comprehensive evaluation because of their balanced design.
* Accuracy provides simple evaluation in most cases, yet imbalanced class distributions make its results unreliable.
* How precisely a model works to prevent incorrect positive predictions determines its precision level.
* The ability to recall depends on detecting actual positive results.
* A F1-Score obtains its measurement by unifying Precision and Recall assessments in a balanced calculation.

**Comparative Analysis:** Every assessment metric confirmed that the combination model provided superior results than single models. The ensemble of CNN with SVM and RF provided the highest performance as CNN demonstrated better results independently while the other models increased overall robustness when CNN performance levels decreased.

**Code Analysis:**

**Preprocessing Steps:**

**1. Imports**

A screenshot of a computer

AI-generated content may be incorrect.

A standard practice exists for importing np for numpy and tf for tensorflow. Due to standard convention sk is used for importing numpy which creates confusion because sk typically represents scikit-learn. The code imports tensorflow using the name "sktf" alongside "sk" for numpy import.

**2. Load the Fashion MNIST dataset**

A close-up of a math formula

AI-generated content may be incorrect.

The Fashion MNIST dataset with images of various garments including shirts and shoes becomes available for processing.

Returns:

* x\_train: Training images (28x28 grayscale)
* y\_train: Corresponding labels
* x\_test, y\_test: Test images and labels

**3. Combine train and test sets**



Merges training and test datasets into a single dataset:

* skx\_data: All images
* sky\_data: All labels

**4. Normalize and reshape the images**



**Normalization:** The model efficiency increases when pixel values in the range of [0, 255] get adjusted to [0, 1].

**Reshaping:** The data undergoes a transformation from (num\_samples, 28, 28) to (num\_samples, 28, 28, 1), aligning with the specific CNN data structure (batch, height, width, channels).

**5. Shuffle the dataset**

****

During training the randomization of sample orders ensures the elimination of potential ordering effects.

**6. Split the data into new training and testing sets**



Splits the combined and shuffled dataset into:

* 70% for training (X\_train, y\_train)
* 30% for testing (X\_test, y\_test)

**Models Steps:**

**1: Build CNN Model**

A screenshot of a computer code

AI-generated content may be incorrect.

Through the Keras API from TensorFlow users can create a Convolutional Neural Network (CNN) within this block. The model implements two convolutional layers with 32 and 64 filter banks alongside max pooling reductions on spatial dimensions. The extracted features are flattened and transmitted to dense hidden layers with 64 neurons which proceed to the output layer containing 10 neurons to classify between the 10 options (classes) through a softmax activation scheme. The model uses the Adam optimizer and sparse categorical crossentropy loss to perform multi-class classification tasks during compilation.

**2: Train CNN**

****

The instantiated CNN receives preprocessed training data for learning operations. Training occurs through 10 epochs at a batch size of 64. During training we reserve 20% of training data for validation to check performance and minimize cases of overfitting.

**3: CNN Predictions**



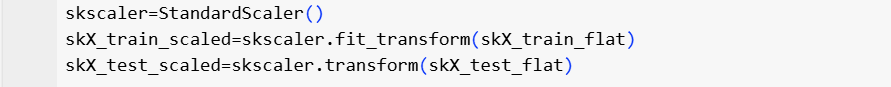
After completion of training the CNN generates probability predictions for each class in the test dataset. A prediction occurs through argmax which chooses the class with the highest predictability for the outcome.

**4: Flatten Images for Traditional Models**

****

SVM together with Random Forest models accept input vectors in one dimension instead of multidimensional images. The block uses a single operation to convert every 28x28 image into a 784-dimensional vector that serves as input for these traditional machine learning algorithms.

**5: Scale Data for SVM**

****

SVMs demonstrate sensitivity toward variations in input features values. StandardScaler transforms the flattened image data by normalizing it to have mean zero and standard deviation one to enhance SVM performance and stabilization.

**6: Train and Predict with SVM**

**A close-up of a computer code

AI-generated content may be incorrect.**

A trained SVM classifier based on an RBF kernel operates on the scaled training data set. By adding probability=True SVM enables both prediction of class labels alongside their respective probabilities. During testing the model provides both class prediction probabilities alongside its final predictions.

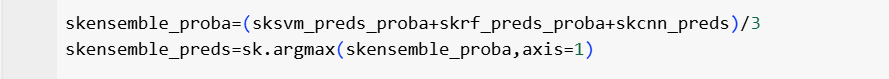
**7: Train and Predict with Random Forest**

**A close-up of a math equation

AI-generated content may be incorrect.**

The training occurs for a Random Forest classifier containing 100 decision trees through utilization of the unscaled flattened images. Random Forest models manage without feature scaling because they operate differently than SVMs. The model generates class probabilities and final class predictions simultaneously on test dataset samples.

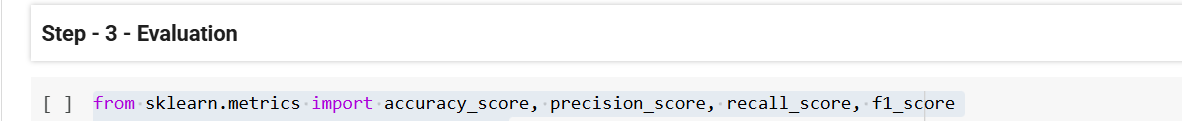
**8: Ensemble the Predictions**

****

The model block performs ensemble prediction using average probabilities derived from CNN, SVM and Random Forest model predictions. Each image from the test set is assigned to the class which possesses the maximum average class probability. Ensemble models enhance their general performance by maximizing the advantages of different modeling techniques.

**Evaluation steps:**

**1.Import Metrics**

****

This line adds four essential classification evaluation metrics from sklearn.metrics to the codebase.

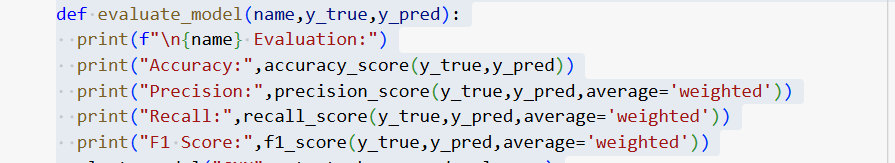
**accuracy score**: Accuracy score counts the percentage of successful predictions against total predictions.

**precision score**: It determines the fraction of correctly identified predicted positive classes in comparison to total predictions. The 'weighted' average addresses label imbalance in its calculations.

**recall score**: The evaluation metric determines the number of actual positive classes that testers correctly identified.

**f1\_score**: A harmonic mean of precision and recall provides a useful measurement when there is a class imbalance.

**2. Evaluation Function:**

****

The function evaluate\_model() takes three arguments consisting of name, y\_true and y\_pred. The function needs a model name parameter (name) along with true labels (y\_true) and predicted labels (y\_pred). The function executes calculations of four essential evaluation metrics before producing print output functions. The weighted value of average='weighted' ensures proper weighting of scores according to each class's distribution of true instances in datasets such as Fashion MNIST. This function creates a standardized method to assess and compare different models’ evaluation metrics.

**3.Evaluate Models:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Each of the previously used four classifiers submits to evaluate\_model() function assessment in this segment.

**CNN:** This evaluation platform analyzes prediction outcomes from the convolutional neural network model.

**SVM:** The support vector machine model receives evaluation of its predictions in this section.

**Random Forest:** The random forest classifier obtains prediction assessments.

**Ensemble:** The system performs an analysis of predictions which emerges from averaging probabilities between CNN, SVM, and Random Forest models.

**Output Screenshots:**

**Preprocessing:**

A screenshot of a computer

AI-generated content may be incorrect.

**Output Explanation:** The Fashion MNIST data sets stream from TensorFlow's public storage to serve your machine learning project. The dataset contains four compressed files which form the basis of its structure. The dataset comprises four files including training images with training labels along with test images and their corresponding test labels. The training images file contains 60,000 grayscale 28x28 pixel images depicting clothing types and matches them to their class identifiers (0–9) in the label file. The test images collection together with their labels consists of ten thousand examples the model needs for validation. Each file is saved locally after its initial download so every subsequent run retrieves information from the cache instead of re-downloading it. The fashion\_mnist.load\_data() function initiation starts the automatic download when it executes for the initial time.

**Models:**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Output Explanation:** The output demonstrates how a Convolutional Neural Network (CNN) model learns through 10 training epochs on the Fashion MNIST dataset. A notification appears first to recommend optimal ways for Keras users to define network input shapes. During training each epoch displays training accuracy and loss in addition to steps completed (613 batches) and validation accuracy and loss. For ten epochs the model exhibited initial training accuracy at 71%, which grew to exceed 94%, demonstrating successful knowledge acquisition. The model shows general data generalization capabilities through a steady decrease of validation loss combined with an increase of validation accuracy from 86% to about 91%. We conducted prediction assessment on 657 batches through the test set in the final line.

**Evaluation:**

A screenshot of a computer program

AI-generated content may be incorrect.

**Output Explanation:** Four predictive models consisting of CNN and SVM and Random Forest and Ensemble operate on the Fashion MNIST dataset and deliver their evaluation outcomes. Results from the CNN model show stable performance with 90.78% accuracy and balanced precision and recall metrics along with an F1 score. The SVM model and Random Forest model achieve comparable results which yield a performance score of 89.17% accuracy and 88.41% accuracy respectively. Through model ensembling the Ensemble system achieves peak performance with an accuracy rate of 91.21% standing above individual model scores. By performing model ensembling the system can utilize the advantages of each method to create enhanced classification metrics.

**Overall Performance Summary:**

The CNN achieves high performance because it obtains spatial features directly from image content. Although SVM and Random Forest demonstrate effective performance they fall behind because they work exclusively with flattened image data that removes essential spatial relations. Your ensemble method effectively leverages the advantages of different approaches by minimizing defects through integration to produce highest levels of precision and F1 score measurement.

**Key Observations:**

**1.Data Download and Preprocessing:**

* A download of Fashion MNIST dataset provided grayscale images from training and testing groups.
* A warning recommends using Input (shape=...) instead of direct input shape usage during the first layer of Sequential models for optimal TensorFlow results.

**2.CNN Training Performance:**

* CNN follows a steep learning progression from 71% accuracy toward reaching 94% training accuracy during epoch 10.
* The training accuracy climbed from approximately 71% to 94% over ten epochs.
* Validation accuracy improves consistently, peaking around \*90.73%\*\*, with stable loss values—indicating minimal overfitting and good generalization.

**3.Model Evaluations:**

**CNN:** A strong baseline performance emerges with ~90.78% accuracy while maintaining precision at an equal level.

**SVM:** The image data processing yielded accuracy results at ~89.18% demonstrating successful though slightly less potent results.

**Random Forest:** The ~88.41% accuracy demonstrates the statistical weakness of this model because it was not optimized to process spatial features.

**Ensemble:** The ensemble approach produces the best overall performance by reaching 91.21% accuracy which indicates that model integration strengthens prediction stability.

**Conclusion:** The experimental results show that deep learning models, especially CNN, achieve excellent performance as image classifiers for Fashion MNIST applications. An ensemble of CNN with traditional machine learning approaches (SVM and Random Forest) results in enhanced performance levels by reaching the most accurate overall outcomes and F1 score rates. The results show that ensemble methods increase both model reliability and prediction accuracy which establishes their importance for dependable real-world systems.