Title: Automated Model Ensemble Techniques for Improved Accuracy

Abstract:

This document presents a study on the implementation and evaluation of automated ensemble learning techniques to improve the accuracy of classification tasks. By utilizing a range of machine learning classifiers such as Logistic Regression, Random Forest, XGBoost, LightGBM, and CatBoost, and combining them using ensemble methods like Voting and Stacking, this research explores how model ensembles can yield better predictive performance. Experiments are conducted on benchmark datasets including the Breast Cancer and Iris datasets.

1. Introduction:

Machine learning models often face challenges in generalization and accuracy due to the variance and bias inherent in individual classifiers. Ensemble techniques aim to overcome these limitations by combining multiple models to produce more robust predictions. This study focuses on two primary ensemble strategies: Voting and Stacking.

2. Methodology:

2.1 Datasets:

- Breast Cancer Dataset: Loaded using sklearn.datasets.load_breast_cancer, suitable for binary classification.
- Iris Dataset: Loaded using sklearn.datasets.load_iris, used for multiclass classification.

2.2 Preprocessing:

- Data standardization was applied using StandardScaler for the breast cancer dataset.
- Train-test split with an 80-20 ratio.

2.3 Models Used:

- Logistic Regression
- Random Forest Classifier
- XGBoost Classifier
- LightGBM Classifier
- CatBoost Classifier
- Gradient Boosting Classifier

2.4 Ensemble Techniques:

- Voting Classifier: A soft and hard voting classifier combining multiple base learners.
- Stacking Classifier: Uses base classifiers to generate predictions which are then used by a meta-classifier (Logistic Regression).

3. Implementation:

3.1 Breast Cancer Dataset:

```
# Ensemble models definition and training
voting_clf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('xgb', clf3)], voting='soft')
voting_clf.fit(X_train, y_train)

stack_clf = StackingClassifier(classifiers=[clf1, clf2, clf3], meta_classifier=LogisticRegression())
stack_clf.fit(X_train, y_train)
```

3.2 Iris Dataset:

```
# Ensemble models on Iris dataset
ensemble = VotingClassifier(estimators=[('lr', model1), ('rf', model2), ('gb', model3)], voting='hard')
```

```
ensemble.fit(X_train, y_train)
```

4. Evaluation Metrics:

- Accuracy
- Classification Report (Precision, Recall, F1-score)
- Confusion Matrix (optional)

5. Results:

The ensemble models generally performed better or comparable to individual models.

Breast Cancer Dataset:

- Voting Classifier Accuracy: ~0.97
- Stacking Classifier Accuracy: ~0.96

Iris Dataset:

- Voting Ensemble Accuracy: ~1.00
- Individual classifiers ranged between 0.93 - 1.00

6. Visualization Dashboard (Streamlit):

A Streamlit dashboard was developed to visualize and compare model performance interactively.

```
st.title("Model Ensemble Dashboard")

st.subheader(name)

st.write(f"Accuracy: **{acc:.2f}**")
```

Run the dashboard using:

```
streamlit run ensemble_dashboard.py
```

7. Conclusion:

Ensemble techniques such as Voting and Stacking provide a practical and effective approach to improve model accuracy and robustness. These methods can be integrated into real-world machine learning pipelines and serve as powerful tools for both binary and multiclass classification tasks.

8. Future Work:

- Integration of automated hyperparameter tuning
- Exploration of bagging and boosting ensemble variations
- Real-time deployment of ensemble models

References:

- Scikit-learn documentation
- XGBoost, LightGBM, CatBoost official documentation
- Streamlit library for visualization