```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier, VotingClassifier
from sklearn.svm import OneClassSVM
from sklearn.ensemble import IsolationForest
from sklearn.metrics import accuracy_score, classification_report

# Load your dataset (replace 'data.csv' with your dataset)
data = pd.read_csv('data.csv')

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(data.drop('label', axis=1), data['label'],
test_size=0.2, random_state=42)

# Anomaly Detection with Isolation Forest
iso_forest = IsolationForest(contamination=0.05, random_state=42)
iso_forest.fit(X_train)
iso_preds = iso_forest.predict(X_test)

# Anomaly Detection with One-Class SVM
svm = OneClassSVM(nu=0.05)
svm.fit(X_train)
svm_preds = svm.predict(X_test)

# Ensemble using Random Forest
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
rf_classifier.fit(X_train, y_train)

# Create a voting ensemble with anomaly detectors and Random Forest
ensemble = VotingClassifier(estimators=[
    ('Isolation Forest', iso_forest),
    ('One-Class SVM', svm),
    ('Random Forest', rf_classifier)
], voting='hard')

ensemble.fit(X_train, y_train)
ensemble_preds = ensemble.predict(X_test)

# Evaluate the models
print("Isolation Forest Results:")
print(classification_report(y_test, iso_preds))

print("One-Class SVM Results:")
print(classification_report(y_test, svm_preds))

print("Ensemble Results:")
print(classification_report(y_test, ensemble_preds))
```