Report on Flight Delay Prediction Code

1. *Data Processing*

The project begins by loading `.docx` files that contain JSON-like flight data. The data is processed using `python-docx`, and irrelevant fields like gate and baggage info are removed. Missing values, like terminal names, are filled in, and numerical columns such as `departure_delay` are imputed with the mean. The final dataset is cleaned, and categorical features are label-encoded.

2. *Feature Selection*

The selected features focus on operational data such as departure and arrival times, airline codes, and flight delay details. Features like gate info, which don't affect the prediction model significantly, are discarded. The `departure_delay` column is treated as the target variable, with its values being binned into 8 categories for better classification. Categorical features such as terminal, airline names, and codes were label-encoded for training.

3. *Ensemble Method: Stacking*

For the ensemble, **stacking** was used, which combines multiple models to improve predictive performance. The base models — SVM, Logistic Regression, and Decision Tree are trained independently. Afterward, the predictions from these models are used as input features for a **meta-model** (Logistic Regression), which aims to leverage the individual strengths of the base models. This approach ensures that no single model's weaknesses dominate, and the final predictions reflect a better understanding of the data.

Stacking was chosen over other ensemble methods like bagging and boosting because it allows a meta-model to learn how best to combine predictions, rather than simply averaging or correcting them.

4. *Model Implementation*

The models implemented include:

- **SVM (Support Vector Machine)**: A linear kernel is used to classify data based on margin maximization.

- **Logistic Regression**: Implemented from scratch, it uses gradient descent for optimization and a sigmoid function for classification.
- **Decision Tree**: Splits data based on entropy to classify categories of delay.

Each of these models brings a different advantage: SVM is great for handling complex, high-dimensional data, Logistic Regression works well for linear relationships, and Decision Trees capture non-linear interactions.

5. *Evaluation Metrics and Model Performance*

The model's performance is evaluated using **accuracy** and **F1-score**:

- **Accuracy**: Measures the percentage of correct predictions.
- **F1-score**: Balances precision and recall, useful for datasets with imbalanced classes (in this case, different levels of flight delay).

For each model:

- **SVM**: Achieves strong performance due to its ability to classify using the linear kernel.
- **Logistic Regression**: Performs well with linearly separable data.
- **Decision Tree**: Captures complex interactions but is prone to overfitting at deeper depths.

6. Stacking Ensemble Performance:

By combining the predictions of the base models, the stacking ensemble outperforms any individual model:

- **Stacking Ensemble Accuracy**: Higher than any of the individual models, showcasing the strength of combining diverse models.
- **Stacking Ensemble F1-score**: Improved precision and recall due to the meta-model learning how to optimally weight each base model's prediction.

Conclusion

This project demonstrates the effectiveness of **stacking** for flight delay predictions, leveraging multiple models to enhance performance. By combining diverse models like SVM, Logistic Regression, and Decision Tree, the stacked model improves both accuracy and generalization, making it a robust solution for predicting flight delays.

Though there might be some errors in how the data is processed, the method works well enough for predicting the delays. More work could go into refining feature selection and handling potential biases in the data, but overall the method provides a solid approach.