1. **Data Preparation**

   Steps Performed:

   - Load CSV into Pandas dataframe and drop duplicate rows in the dataframe
   - Inspect the dataframe and find columns “Name”, “Outcome Type”, and “Outcome Subtype” have missing values. Group cells by features “breed”, “color”, “sex upon outcome”, and “age upon outcome”. Replace missing values in the group with the most frequent corresponding values in the group.
   - Convert “DateTime” to 4 features “year”, “month”, “day”, and “hour” each with numerical type. Convert “Date of Birth” to “dob_year”, “dob_month”, and “dob_day”. Transform “Age upon Outcome” into a single numeric variable “age_outcome_days”.
   - For high-cardinality features such as “Breed”, “Color”, and “Name”, frequency encoding was applied so that they were replaced by numerical columns indicating how often each value appeared.
   - For other features, one-hot encoding was applied with pd.get_dummies().
   - After processing, the dataset contained 31 features mixed with numeric and categorical. Irrelevant columns such as “Animal ID” and “Breed_freq” were dropped.

2. **Insights from Data Preparation**

   - **Outcome Distribution:** There were two classes -- “Adoption” and “Transfer”. The size of “Adoption” class is twice the size of “Transfer” class representing an unbalanced dataset.
   - **Animal Type:** Most animals are dogs and cats.
   - **Sex Distribution:** The most common categories were Spayed Female and Neutered Male, suggesting most adopted animals are sterilized.

These insights indicate that there’s a class imbalance in the dataset and thus it’s easier for models to predict the output as “Adoption”.

3. **Model Training Procedure**

These classification models were trained to predict “Outcome Type” (Adoption = 1, Transfer = 0), split the dataset with 30% testing set and 70% training set, and also ensure class balance in both sets.

   - **K_Nearest Neighbors(KNN)**
     - Initial model with n_neighbors = 300
     - Trained using fit() and evaluated on the test set
   - **KNN with GridSearchCV**
     - Hyperparameter tuning over n_neighbors {5,10, 25, 50, 100, 200, 300, 400} using 5-fold cross-validation.
     - Use the best estimator to predict on the test set.
   - **Linear Classification (SGDClassifier)**
     - Used a perceptron loss with alpha=0.05
     - Designed to model a linear decision boundary efficiently

All models were evaluated using Accuracy, Precision, Recall, and F1-score to capture overall, positive-predictive, sensitivity, and balanced performance.

4. Model Performance

Model	Accuracy	Precision	Recall	F1-score
KNN (300)	0.7364	0.7350	0.9173	0.8161
KNN (GridSearchCV)	**0.7626**	**0.7602**	0.9169	**0.8312**
Linear Classifier	0.6533	0.6516	**0.9805**	0.7829

- The **GridSearchCV KNN** performed best overall, with the highest accuracy and F1-score, showing a balanced trade-off between precision and recall.
- The **Linear Classifier** achieved the highest **recall** (0.98), meaning it captured nearly all positive (adoption) cases, but at the cost of lower precision and accuracy.
- The **KNN models** provided more consistent, generalizable predictions.

5. Confidence in the Model

The chosen model (KNN with GridSearchCV) exhibits good predictive performance with an F1-score of 0.83, reflecting a strong balance between precision and recall. Given the class imbalance and high recall, I'm confident that the model effectively identifies adoptions.

In practice, recall is the most important metric because animal shelters should allocate resources and prepare adoption events for adopted animals. If the model misses adoptions, those animals might be wrongly predicated as “Transfer” and not receive proper attention.

All selected model provides high sensitivity with strong overall accuracy, making them reliable and trustworthy for decision support in this dataset.