Prepared by group A

Group Project

Early Diabetes Risk Screening

Using KNN Classifier, Logistics Regression, Support Vector Machine

30 August 2025

Team Members

Aye Chan Myint

Mentor

Ei Ei Tun

Member

Aint Kyi Phyu Shin

Member

Aye Charm Ko Ko

Member

Introduction – The Problem & Importance

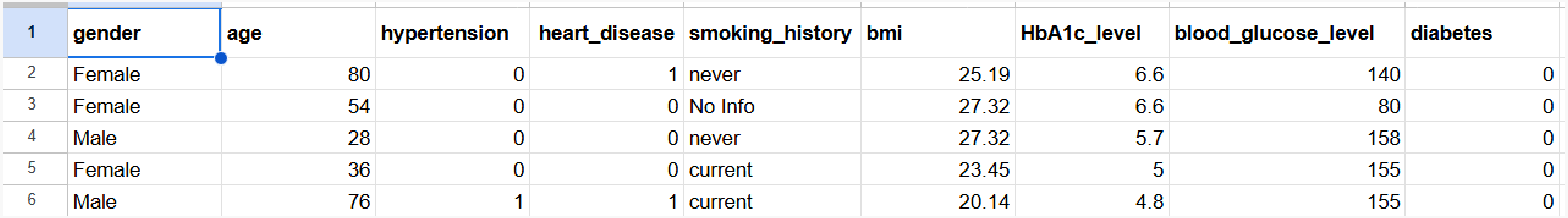
📌 Classification Problem

- Many people remain undiagnosed with diabetes until symptoms are advanced
- Late diagnosis → disease progression & serious complications
- Diabetes is a major global health concern, affecting millions worldwide

🌍 Why It Matters

- Early detection enables:
  - Timely medical intervention
  - Lifestyle changes
  - Reduced long-term complications

Dataset Description

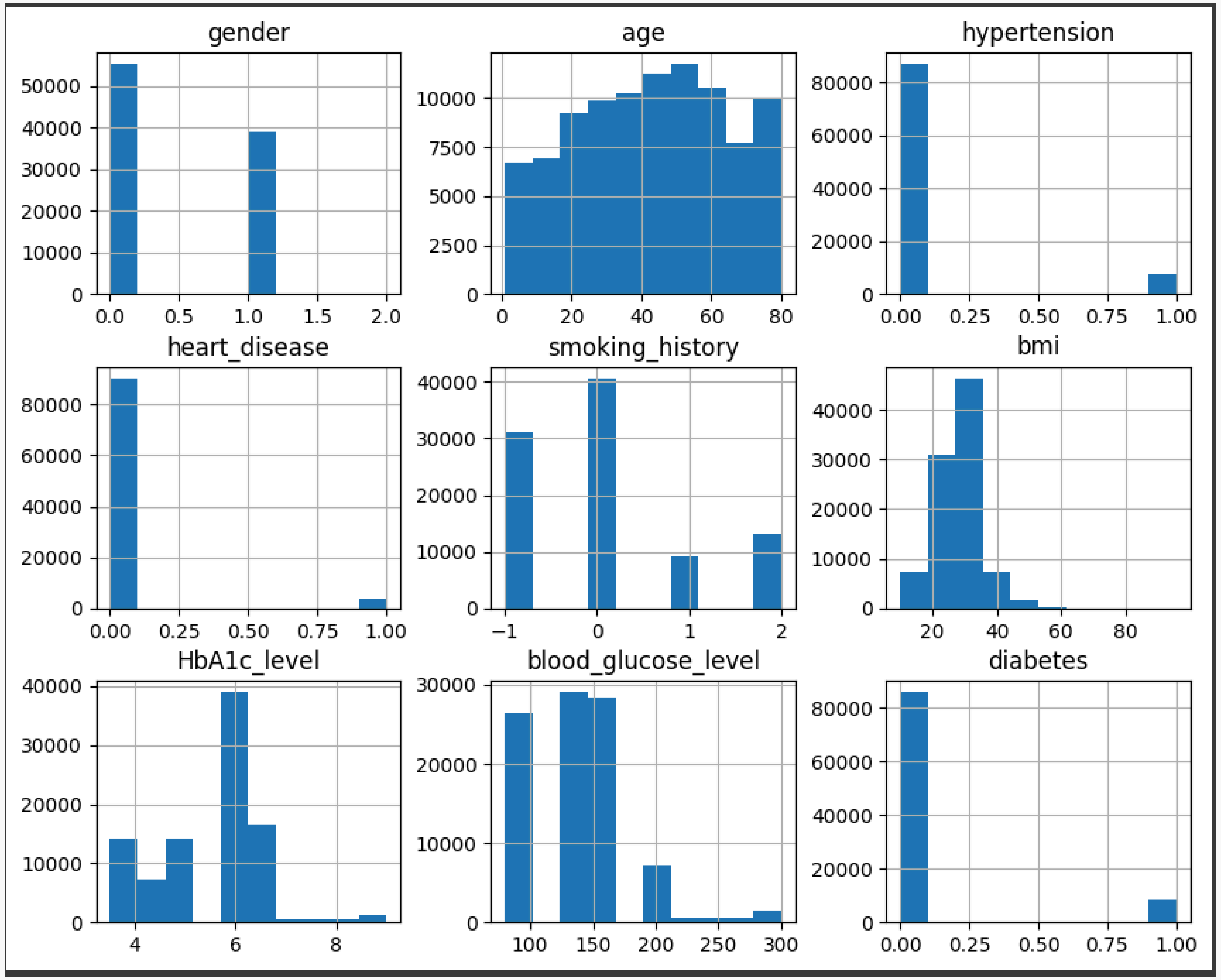


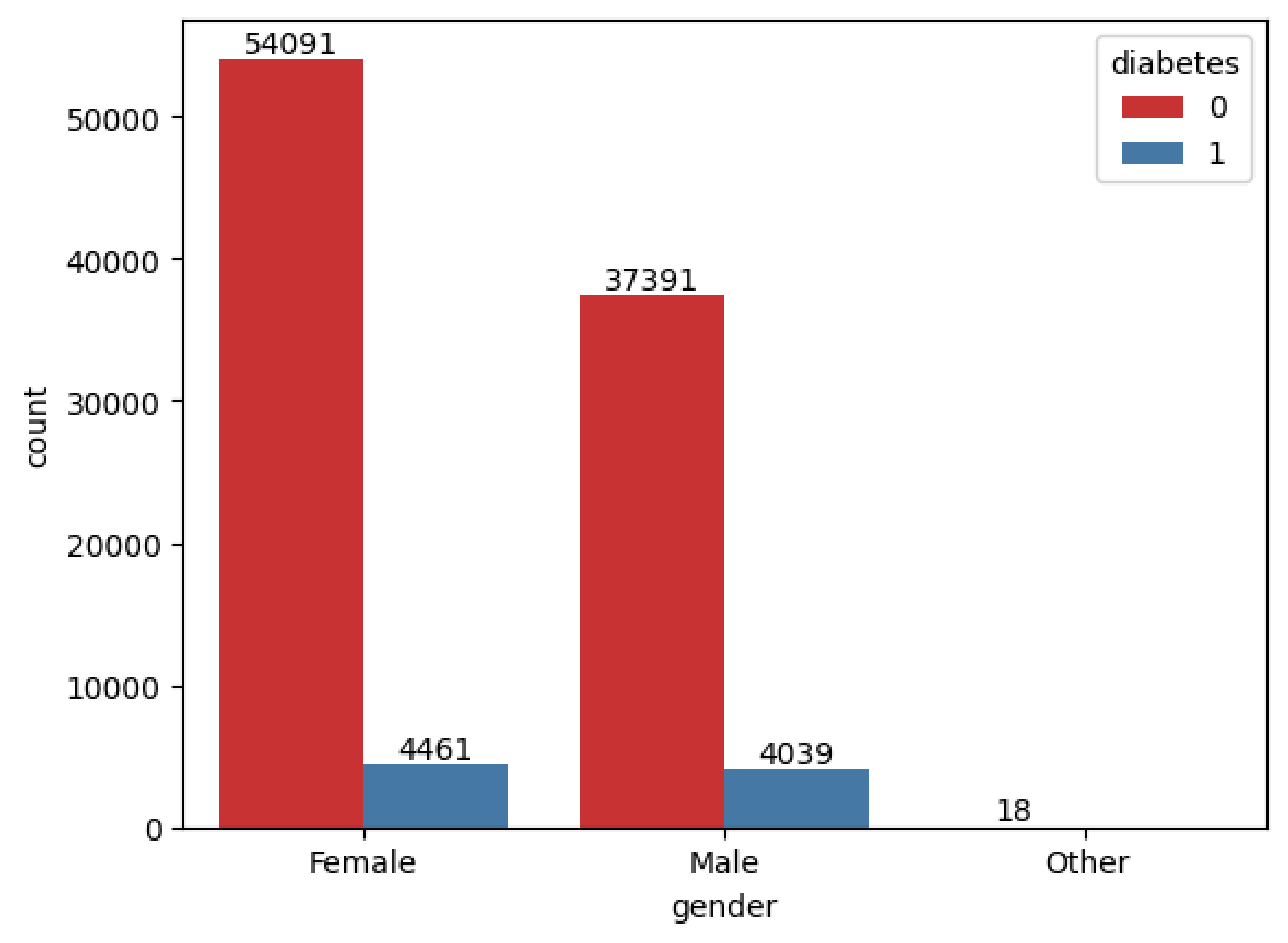
1	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
2	Female	80	0	1	never	25.19	6.6	140	0
3	Female	54	0	0	No Info	27.32	6.6	80	0
4	Male	28	0	0	never	27.32	5.7	158	0
5	Female	36	0	0	current	23.45	5	155	0
6	Male	76	1	1	current	20.14	4.8	155	0

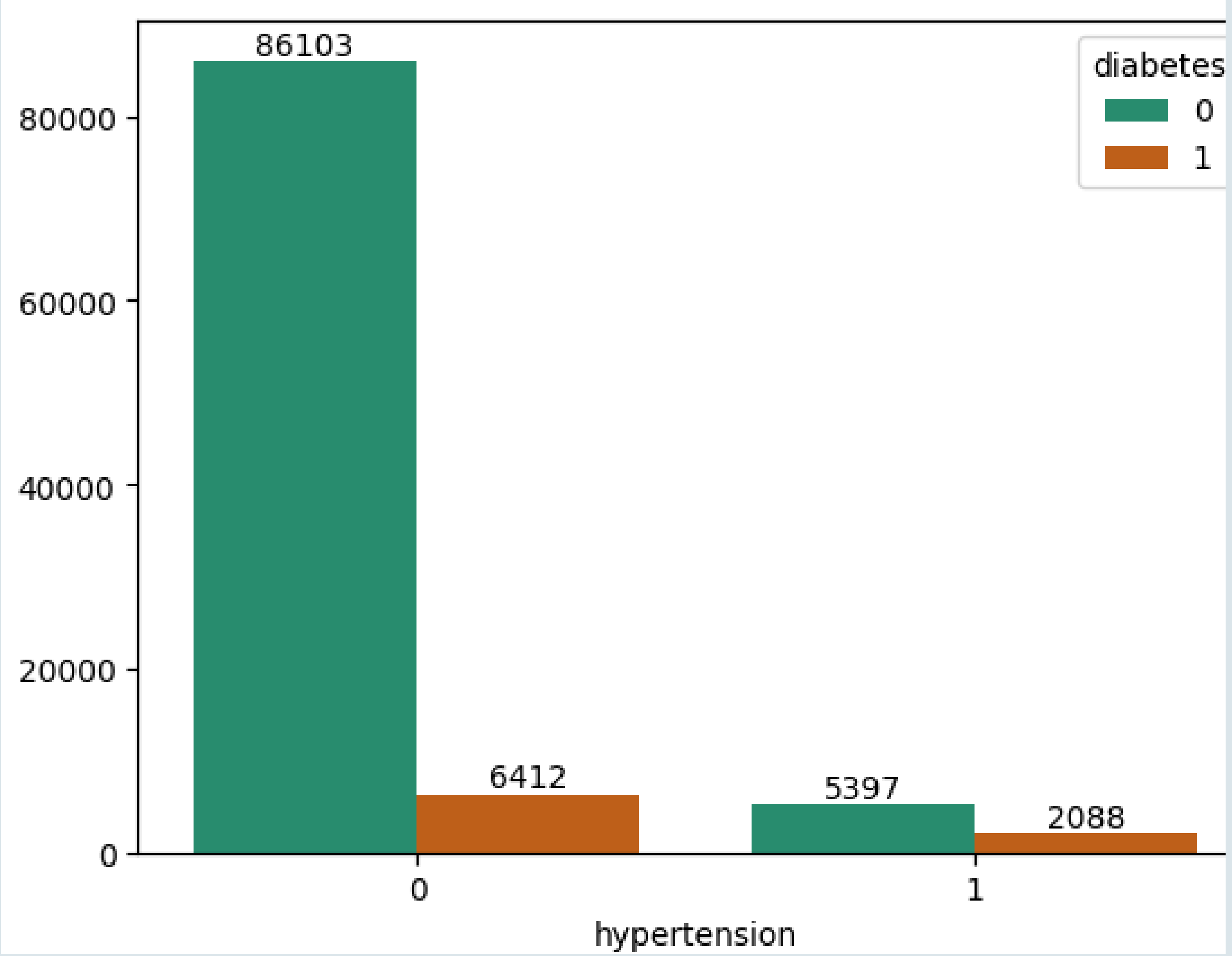
Early Diabetes Risk Screening Analysis

- Dataset: Pima Indians Diabetes Dataset (UCI Repository)
- Samples: 100,000 patients
- Features: 9 total
  - Numeric: Age, BMI, HbA1c_level, Blood_glucose_level
  - Categorical: Gender, Smoking_history
- Target Variable: Diabetes (binary → 0 = no, 1 = yes)
- Note: Some features have biologically impossible values (e.g., BloodPressure = 0) → treated as missing & imputed

Early Diabetes Risk Screening Analysis





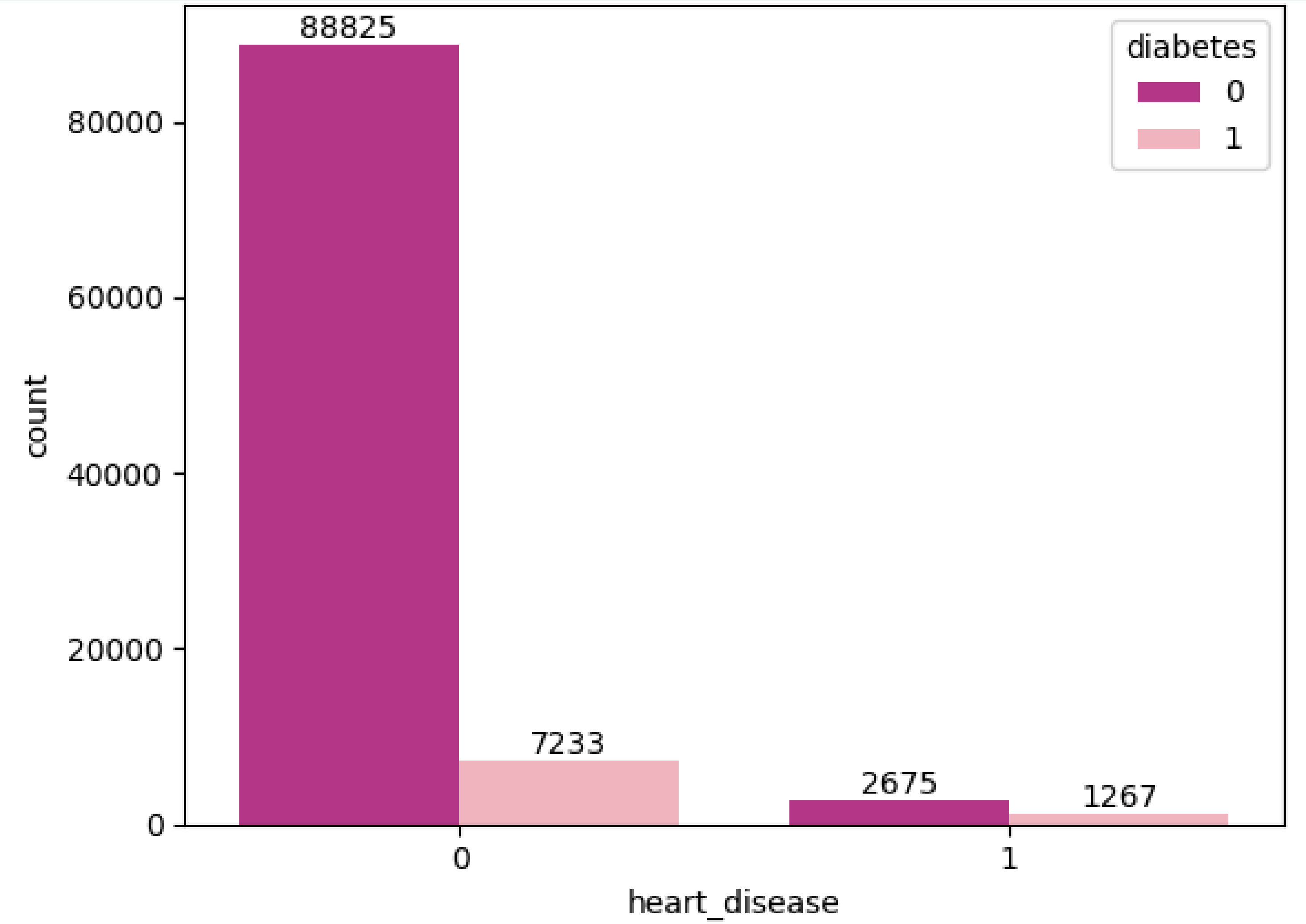


Gender

- Show the total number of males and females in the dataset.
- Compare diabetes cases between males and females to see trends.
- Understand if the dataset is balanced by gender for modeling purposes.

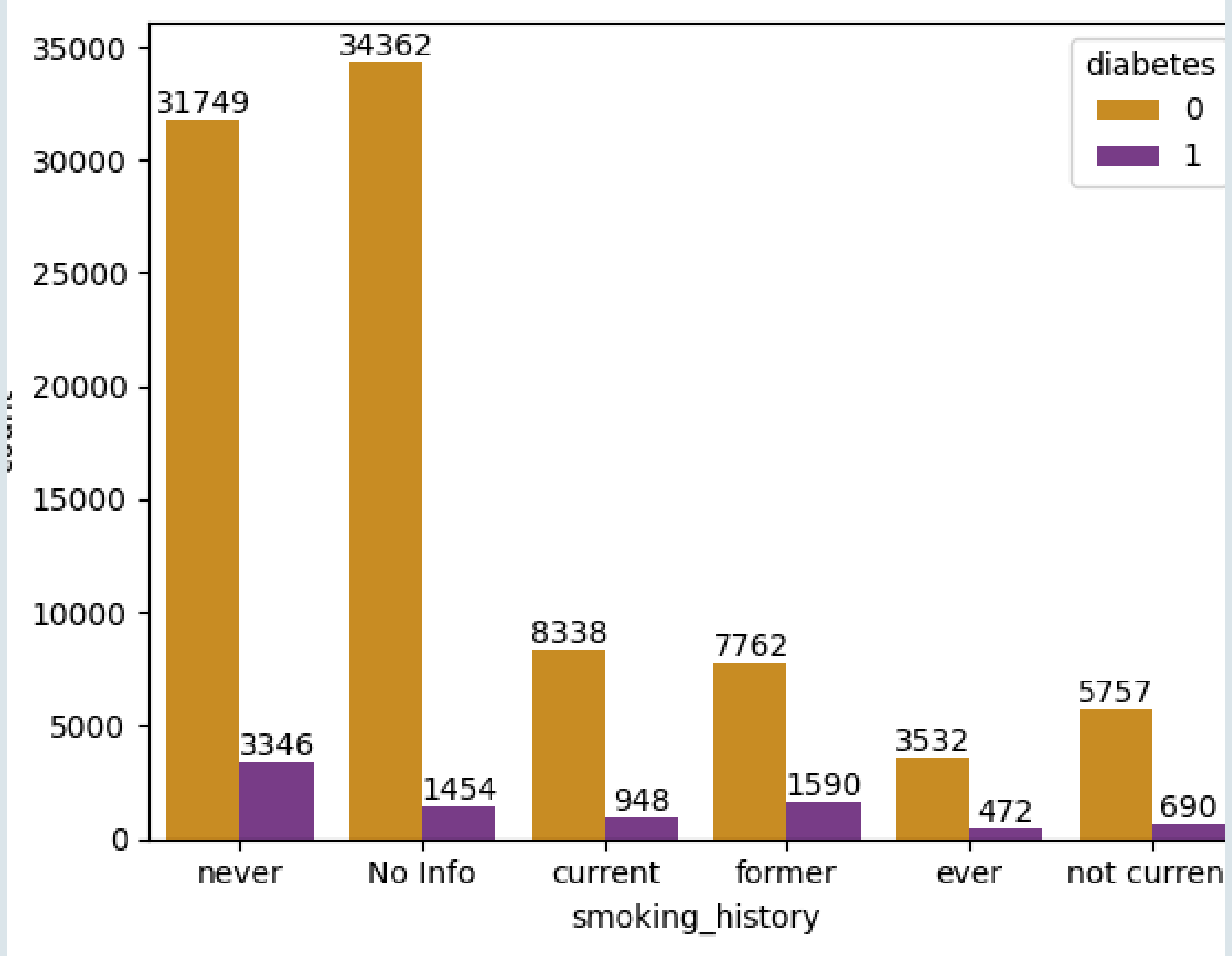
Hypertension

- Visualize the relationship between hypertension and diabetes.
- Identify if patients with hypertension have a higher likelihood of diabetes.
- Highlight patterns that could help in risk analysis and prediction.



Heart Disease

- Visualize the relationship between heart disease and diabetes.
- Compare diabetes cases in patients with and without heart disease.
- Identify if heart disease is a risk factor for diabetes.



Smoking History

- Examine the link between smoking history and diabetes.
- Compare diabetes prevalence among different smoking categories.
- Highlight patterns for lifestyle-related diabetes risk.

Data Preprocessing

Data Cleaning

- Replaced biologically impossible zeros (e.g., BloodPressure = 0)
- Imputed missing value**s**

🔤 Categorical Encoding

- Gender → Label Encoding (Male = 1, Female = 0)
- Smoking History → Mapped to numerical values
    - Never = 0
    - No Info = -1
    - Current = 2
    - Former = 1
    - Ever = 2
    - Not Current = 0

Data Preprocessing

🔢 Data Type Conversion

- Converted Age column → integer type

📊 Feature Scaling

- Standardized numeric features using StandardScaler

✅ Feature Selection

- Retained relevant predictors:
  - Age, BMI, HbA1c_level, Blood_glucose_level, Gender, Smoking_history
- Removed redundant/uninformative features

Model Selection & Justification

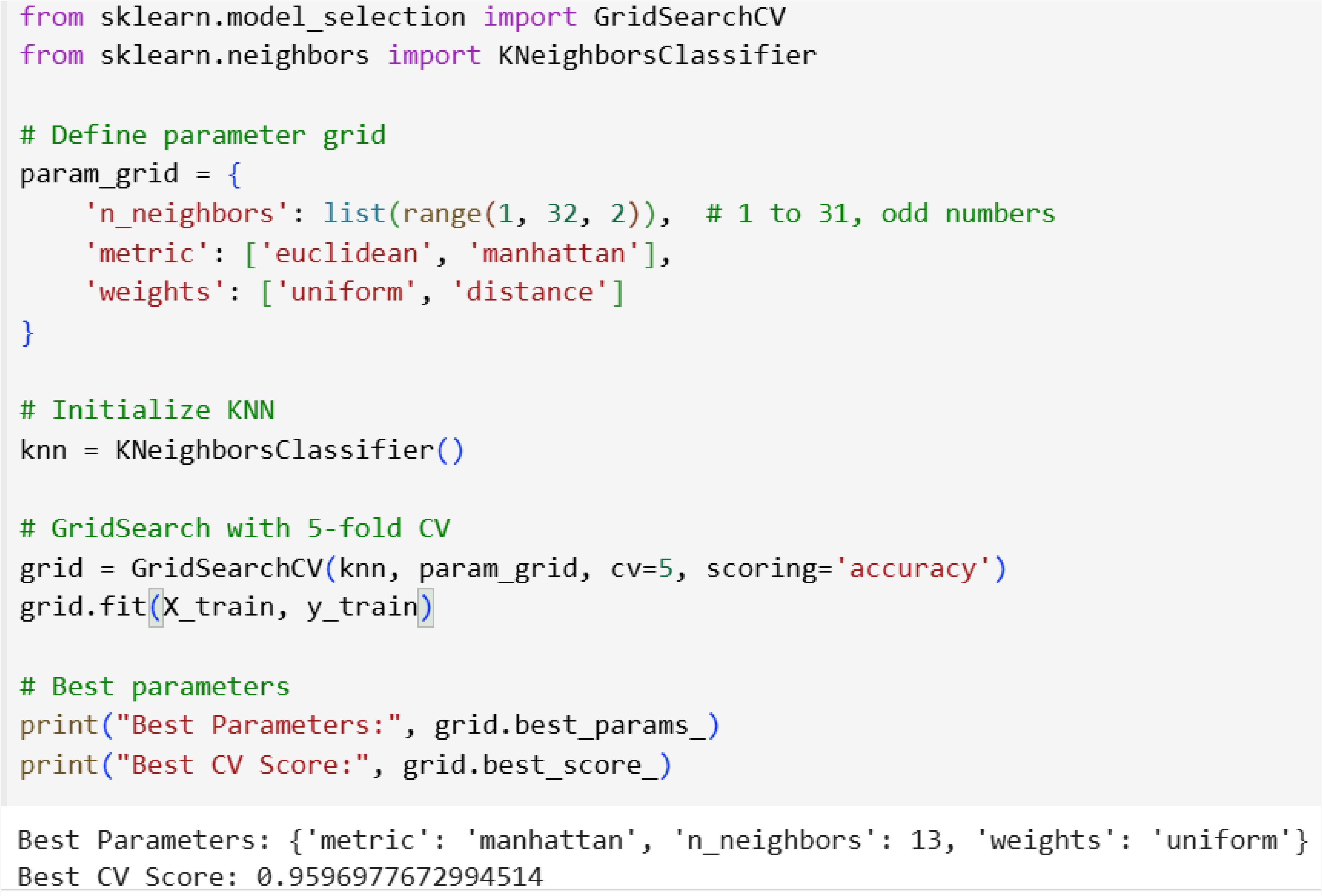
Chosen Models: KNN, SVM, Logistic Regression (LR)

- **KNN**
  - Works with numeric features: age, BMI, HbA1c, blood_glucose_level
  - Categorical features (gender, smoking_history) encoded
  - Simple, interpretable; classifies patients by similarity
- **SVM**
  - Handles complex boundaries in numeric + encoded categorical data
  - Effective in high-dimensional feature spaces
  - Robust predictive accuracy
- **Logistic Regression**
  - Provides baseline for comparison
  - Interpretable coefficients
  - Shows feature impact on diabetes prediction

Model Implementation

- KNN’s performance depends on the choice of hyperparameters (k, distance metric, and weighting scheme).
- KNN with hyperparameters:
    - Number of neighbors (k = 1–31)
    - Distance metric (Euclidean, Manhattan)
    - Weighting scheme (uniform, distance-based)
- Use GridSearchCV with 5-fold cross-validation.

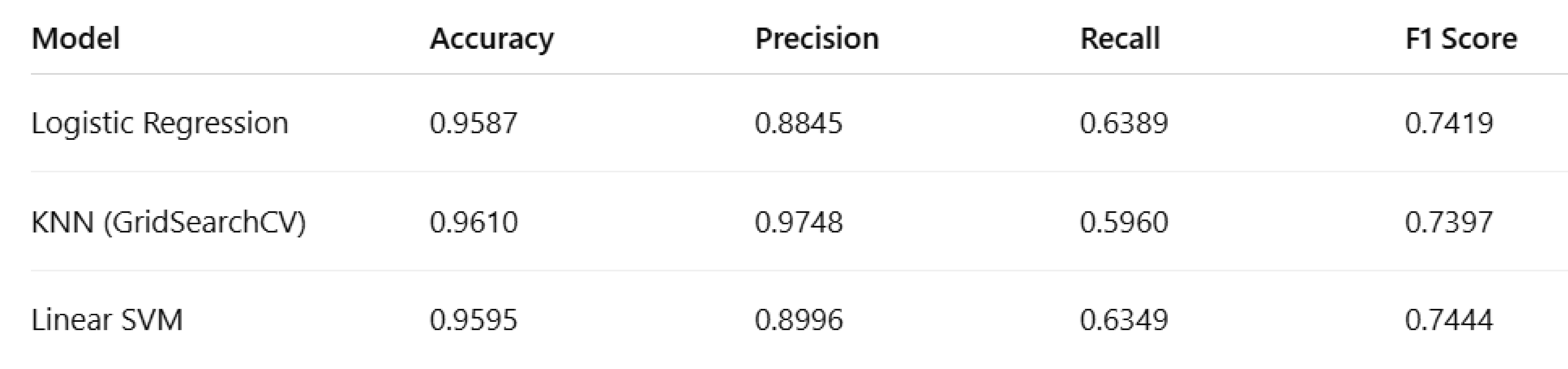
Model Implementation



```python
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier

# Define parameter grid
param_grid = {
    'n_neighbors': list(range(1, 32, 2)),  # 1 to 31, odd numbers
    'metric': ['euclidean', 'manhattan'],
    'weights': ['uniform', 'distance']
}

# Initialize KNN
knn = KNeighborsClassifier()

# GridSearch with 5-fold CV
grid = GridSearchCV(knn, param_grid, cv=5, scoring='accuracy')
grid.fit(X_train, y_train)

# Best parameters
print("Best Parameters:", grid.best_params_)
print("Best CV Score:", grid.best_score_)
```

```
Best Parameters: {'metric': 'manhattan', 'n_neighbors': 13, 'weights': 'uniform'}
Best CV Score: 0.9596977672994514
```

Experimental Results



Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.9587	0.8845	0.6389	0.7419
KNN (GridSearchCV)	0.9610	0.9748	0.5960	0.7397
Linear SVM	0.9595	0.8996	0.6349	0.7444

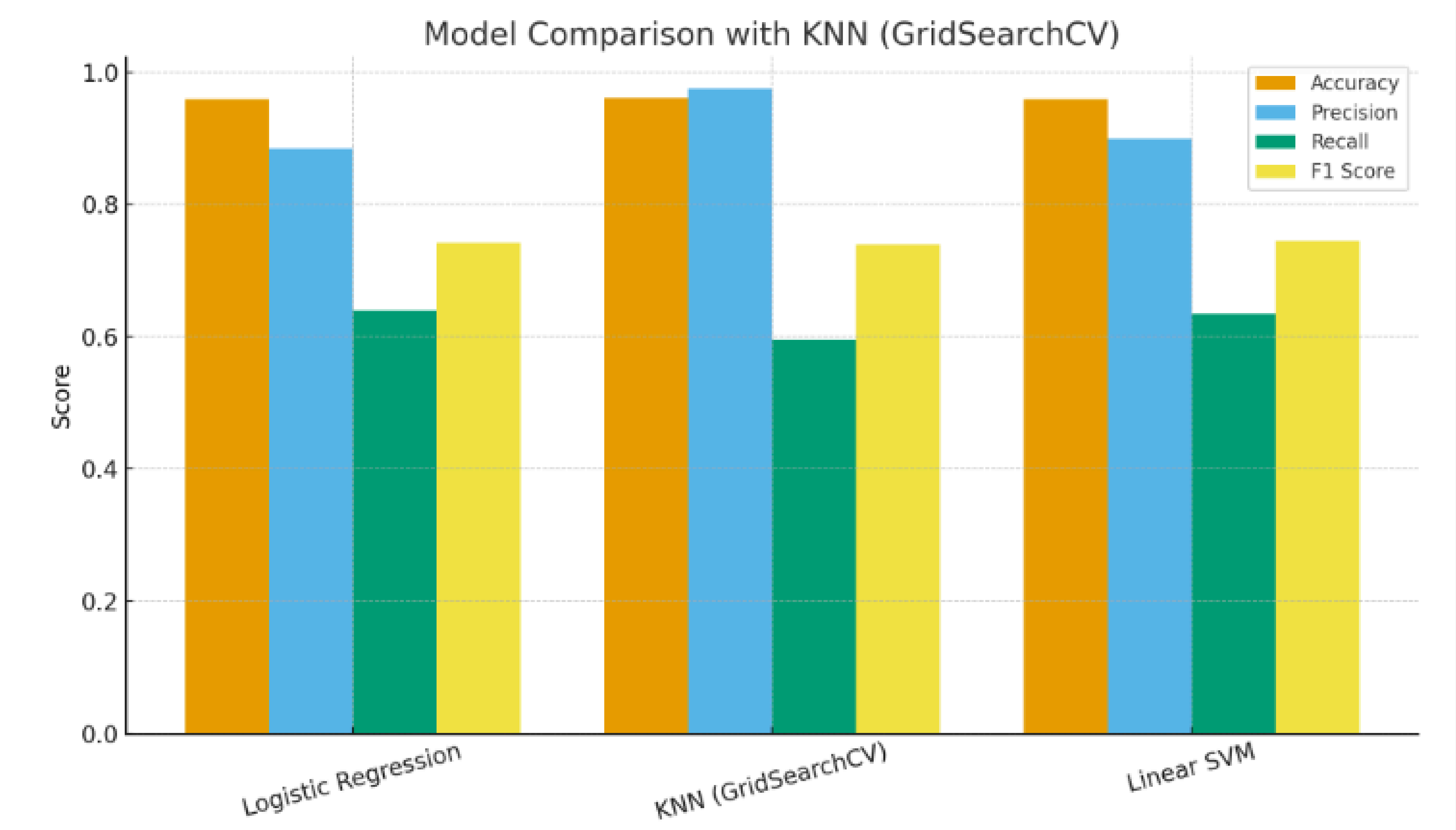
KNN (GridSearchCV):

- Achieves the highest accuracy (0.9610) and extremely high precision (0.975) → almost no false alarms.
- However, recall drops to 0.596, which misses more positives.

Logistic Regression: Better recall (0.639), but slightly lower accuracy and precision.
Linear SVM: Still the best balance overall (highest F1 Score).

Use KNN (GridSearchCV) if avoiding false positives is critical.
Use Logistic Regression if catching positives is more important.
Use Linear SVM if want a balanced trade-off.

Experimental Results

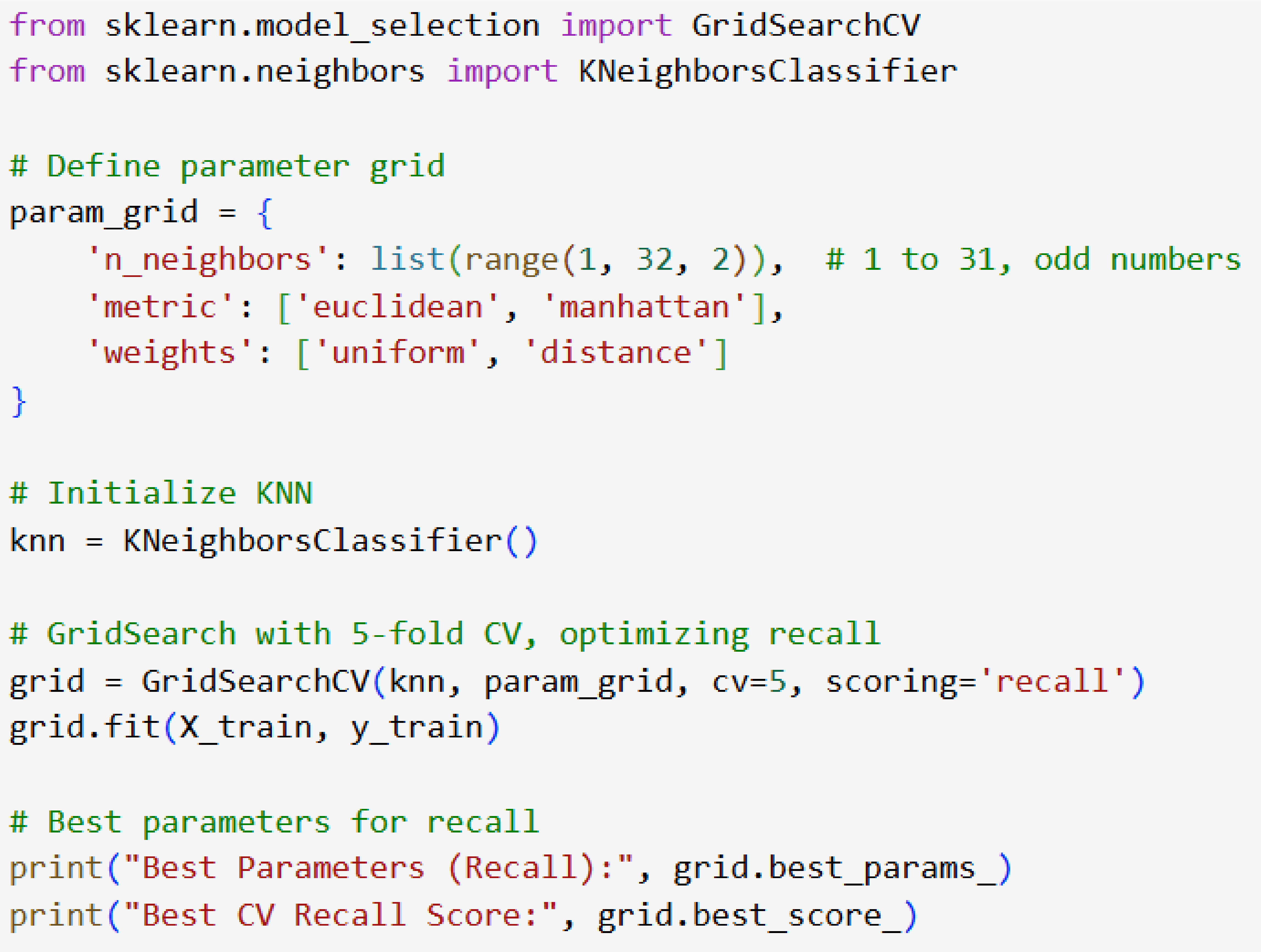


- KNN (GridSearchCV) peaks in accuracy and especially precision, but recall is the lowest.
- Logistic Regression keeps stronger recall.
- Linear SVM stays the most balanced with the best F1 score.

Discussion of Results

- In medical prediction (like diabetes):
  - False negatives (FN) are very costly because missing a diabetic patient means they don’t get treatment.
  - This means Recall is the most important metric, since we want to catch as many true positives as possible.
  - Precision is still relevant, but in healthcare, it’s generally more acceptable to have some false alarms (FP) than to miss real cases.
- Based on results:
  - Logistic Regression: Recall = 0.639 → best at catching positives.
  - Linear SVM: Recall = 0.635 → very close to Logistic Regression, slightly lower.
  - KNN (GridSearchCV): Recall = 0.596 → misses more positive cases, despite high accuracy and precision.

A recall-optimized GridSearchCV



```python
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier

# Define parameter grid
param_grid = {
    'n_neighbors': list(range(1, 32, 2)),  # 1 to 31, odd numbers
    'metric': ['euclidean', 'manhattan'],
    'weights': ['uniform', 'distance']
}

# Initialize KNN
knn = KNeighborsClassifier()

# GridSearch with 5-fold CV, optimizing recall
grid = GridSearchCV(knn, param_grid, cv=5, scoring='recall')
grid.fit(X_train, y_train)

# Best parameters for recall
print("Best Parameters (Recall):", grid.best_params_)
print("Best CV Recall Score:", grid.best_score_)
```

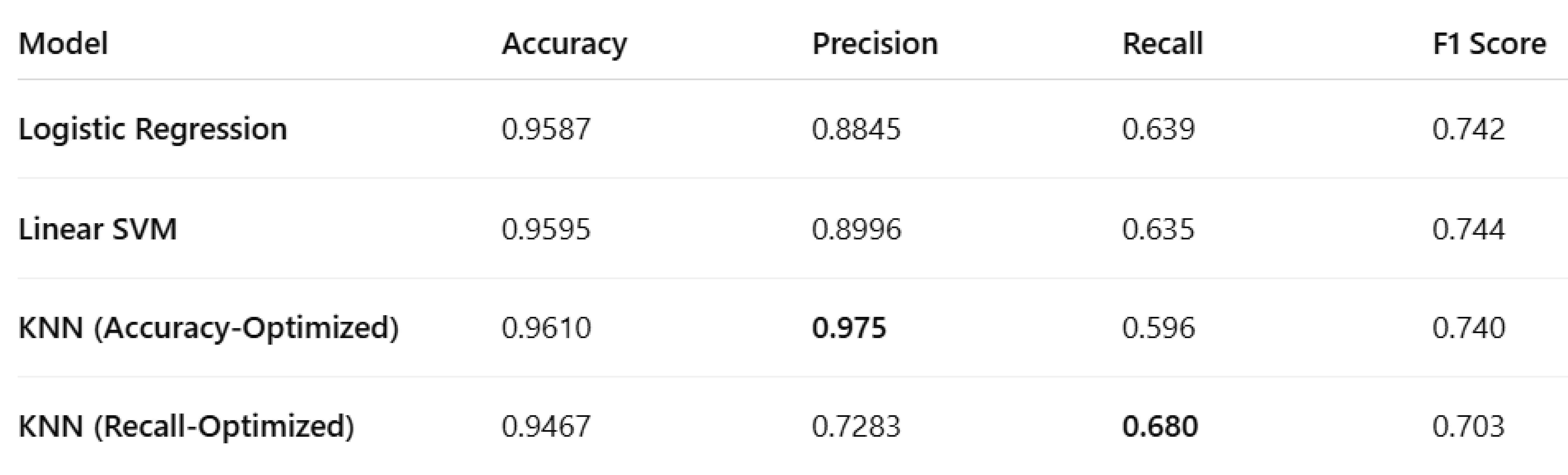
Discussion of Results

What Changed vs. Accuracy-Optimized KNN

- Recall improved (because FN dropped from ~693 → 560, TP rose).
- Precision decreased (more FP = more false alarms).
- Accuracy dropped slightly (because we’re prioritizing recall, not overall balance).

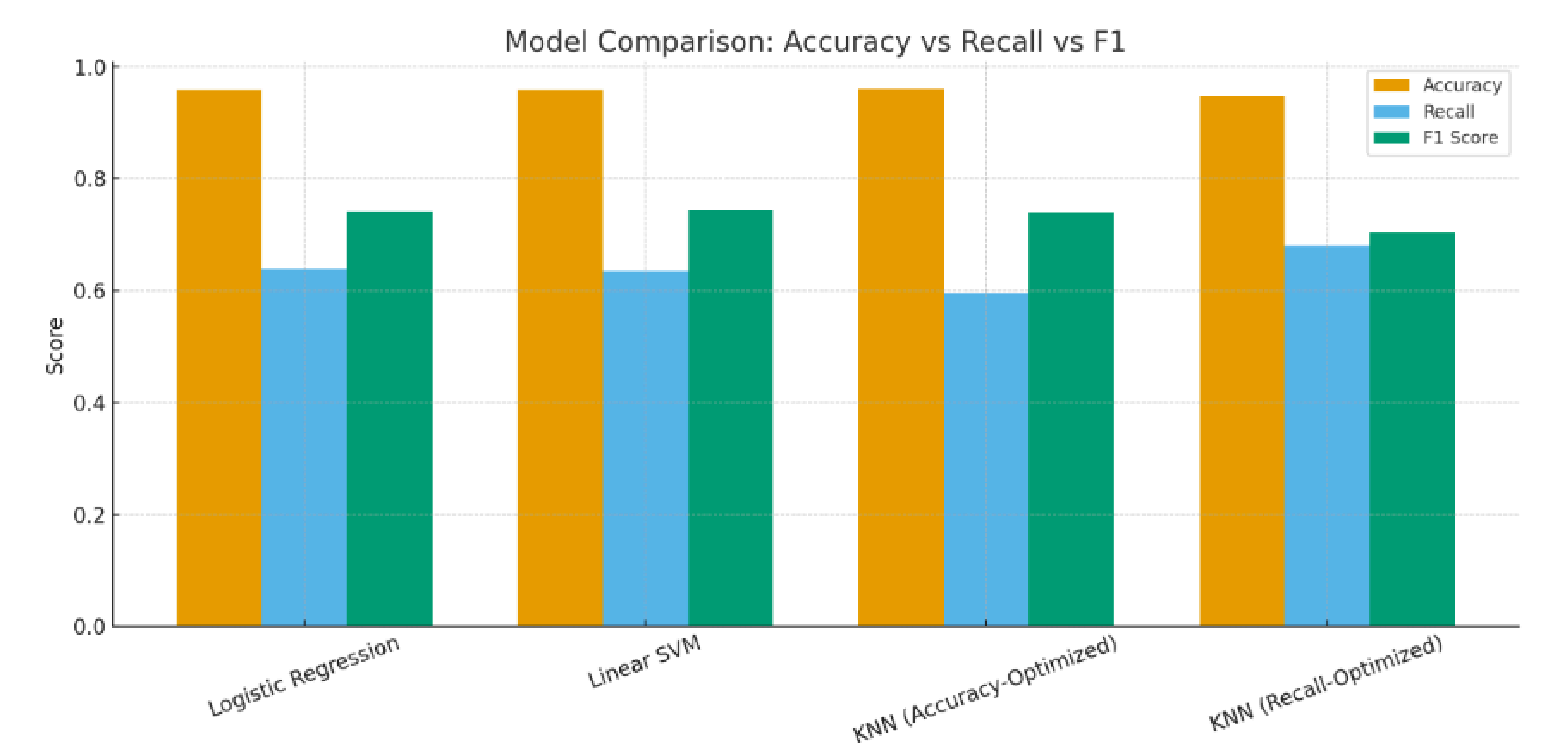
In Medical Context (Diabetes Prediction)

- a good trade-off:
  - Better recall = fewer missed diabetic patients.
  - Acceptable even if some non-diabetic patients are flagged (they just undergo further testing).
- The recall-optimized KNN is now closer to what doctors would prefer in a screening tool.



Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.9587	0.8845	0.639	0.742
Linear SVM	0.9595	0.8996	0.635	0.744
KNN (Accuracy-Optimized)	0.9610	**0.975**	0.596	0.740
KNN (Recall-Optimized)	0.9467	0.7283	**0.680**	0.703

Discussion of Results



Insights

- Precision king: KNN (accuracy-optimized) – catches mostly true diabetics when it predicts positive, but misses many (low recall).
- Recall king: KNN (recall-optimized) – detects more diabetics, but at the cost of many false alarms.
- Best balance: Logistic Regression / Linear SVM – stable F1, strong precision, and reasonable recall.

Conclusion and Future Work

- ML models (KNN, Logistic Regression, SVM) effectively predict diabetes risk.
    - Logistic Regression & SVM → best balance between precision and recall.
    - KNN → tunable for either high precision (fewer false positives) or high recall (catch more true cases).
- In healthcare, recall is critical to avoid missed diagnoses.
- Models show promise as non-invasive, cost-effective screening tools.
- Future work:
    - Expand datasets for broader generalizability.
    - Add lifestyle/clinical features (diet, activity, genetics).
    - Explore advanced models (ensemble, deep learning).
    - Tune thresholds to maximize recall.

Thank you