Telco Customer Churn Prediction

A Complete Machine Learning Pipeline

Machine Learning Project Report

January 2026

Abstract

This report presents a complete machine learning solution for predicting customer churn in the telecommunications industry. We walk through every step from data exploration to model deployment, making it easy for readers to understand and replicate. The project includes four machine learning models, a REST API, and a web interface. Key finding: tenure (customer lifespan) and contract type are the strongest predictors of churn.

Contents

1 Introduction **3**
- 1.1 What is Customer Churn? . . . 3
- 1.2 Our Goal . . . 3
- 1.3 Dataset Used . . . 3

2 Exploring the Data (EDA) **3**
- 2.1 Understanding the Target Variable . . . 3
- 2.2 What Makes Customers Leave? . . . 4
- 2.3 Numeric Features Analysis . . . 5

3 The Machine Learning Pipeline **5**
- 3.1 Step 1-2: Data Cleaning . . . 6
- 3.2 Step 3: Feature Engineering . . . 7
- 3.3 Step 4: Train/Test Split . . . 7
- 3.4 Step 5: SMOTE (Handling Class Imbalance) . . . 7
- 3.5 Step 6: Encoding and Scaling . . . 8

4 Evaluation Metrics Explained **8**

5 Model Training and Results **8**
- 5.1 Models We Tested . . . 8
- 5.2 Results Comparison . . . 9
- 5.3 Confusion Matrices . . . 10
- 5.4 Which Model is Best? . . . 10
- 5.5 What Features Matter Most? . . . 11

6 Deployment **11**
- 6.1 REST API (FastAPI) . . . 11
- 6.2 Web Interface (Streamlit) . . . 11

7 Lessons Learned: Data Leakage **12**

8 Business Recommendations **12**

9 Conclusion **12**
9.1 What We Achieved 12
9.2 Future Improvements 12

1 Introduction

1.1 What is Customer Churn?

Customer churn happens when a customer stops using a company's services. In the telecom industry, this means a customer cancels their phone or internet subscription and moves to a competitor.

Why does it matter?

- Getting a new customer costs 5-7 times more than keeping an existing one
- A 5% increase in customer retention can boost profits by 25-95%
- Early prediction allows companies to offer incentives before customers leave

1.2 Our Goal

We want to build a machine learning model that can predict which customers are likely to leave. This is a **binary classification** problem:

- **Input:** Customer information (demographics, services, billing)
- **Output:** Will this customer churn? (Yes or No)

1.3 Dataset Used

We used the **IBM Telco Customer Churn** dataset with:

- 7,043 customers
- 21 features per customer
- 26.5% of customers actually churned (left the company)

2 Exploring the Data (EDA)

2.1 Understanding the Target Variable

Our target variable is **Churn** with two values:

- **No** (stayed): 5,174 customers (73.5%)
- **Yes** (left): 1,869 customers (26.5%)

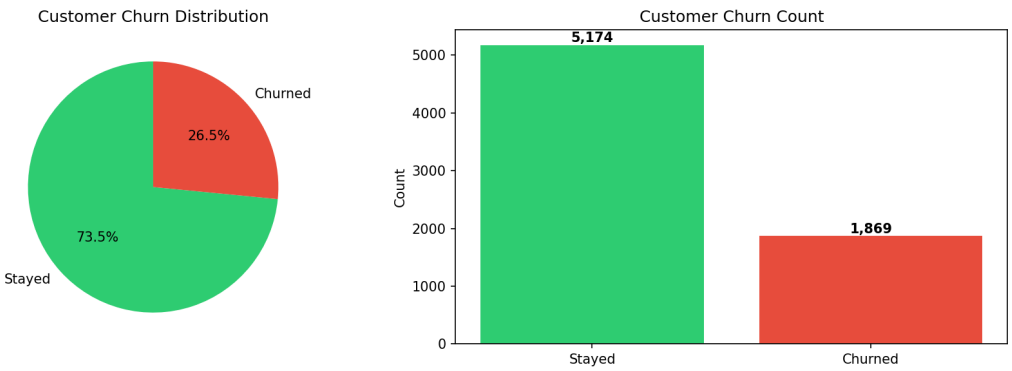

Figure 1: The data is imbalanced—most customers stayed. This is typical in real-world churn scenarios.

Important note: Because only 26.5% of customers churned, a model that always predicts "No churn" would be 73.5% accurate but completely useless! This is why we use metrics like F1-Score and Recall instead of just Accuracy.

2.2 What Makes Customers Leave?

We analyzed various factors to understand what drives churn:

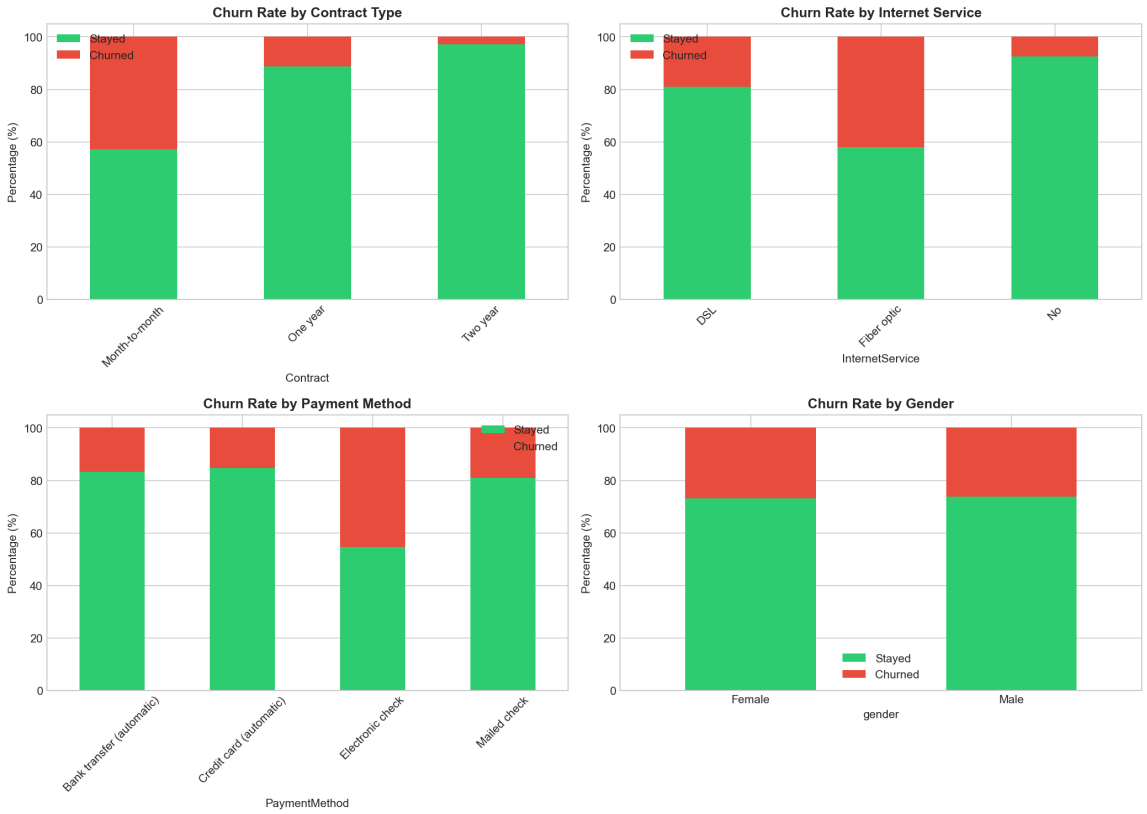

Figure 2: Churn rates vary significantly across different customer segments.

Key discoveries:

1. **Contract type matters most:**
   - Month-to-month contracts: 43% churn rate (very high!)
   - One-year contracts: 11% churn rate
   - Two-year contracts: only 3% churn rate
2. **Payment method affects churn:**
   - Electronic check: 45% churn (highest!)
   - Automatic payments: around 15% churn
3. **New customers are at highest risk:**
   - Customers in their first year: 47% churn
   - Customers with 6+ years: only 7% churn

2.3 Numeric Features Analysis

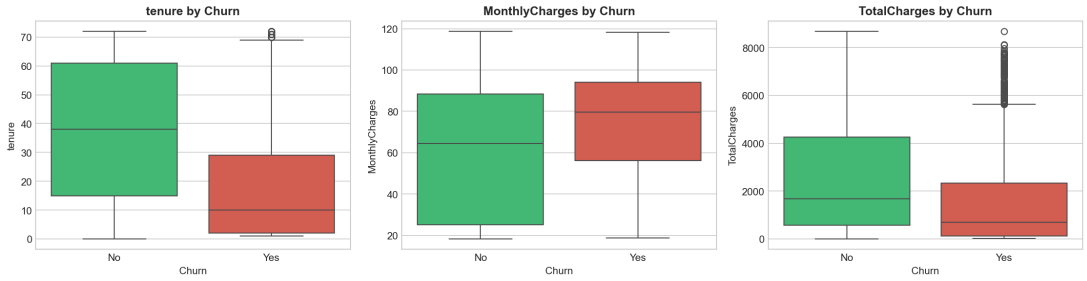

Figure 3: Comparing numeric features between customers who stayed vs. left.

What we learned:

- Churners have **shorter tenure** (fewer months with the company)
- Churners pay **higher monthly charges** on average
- Churners have **lower total charges** (because they left early)

3 The Machine Learning Pipeline

Below is our complete pipeline showing how data flows from raw input to final prediction:

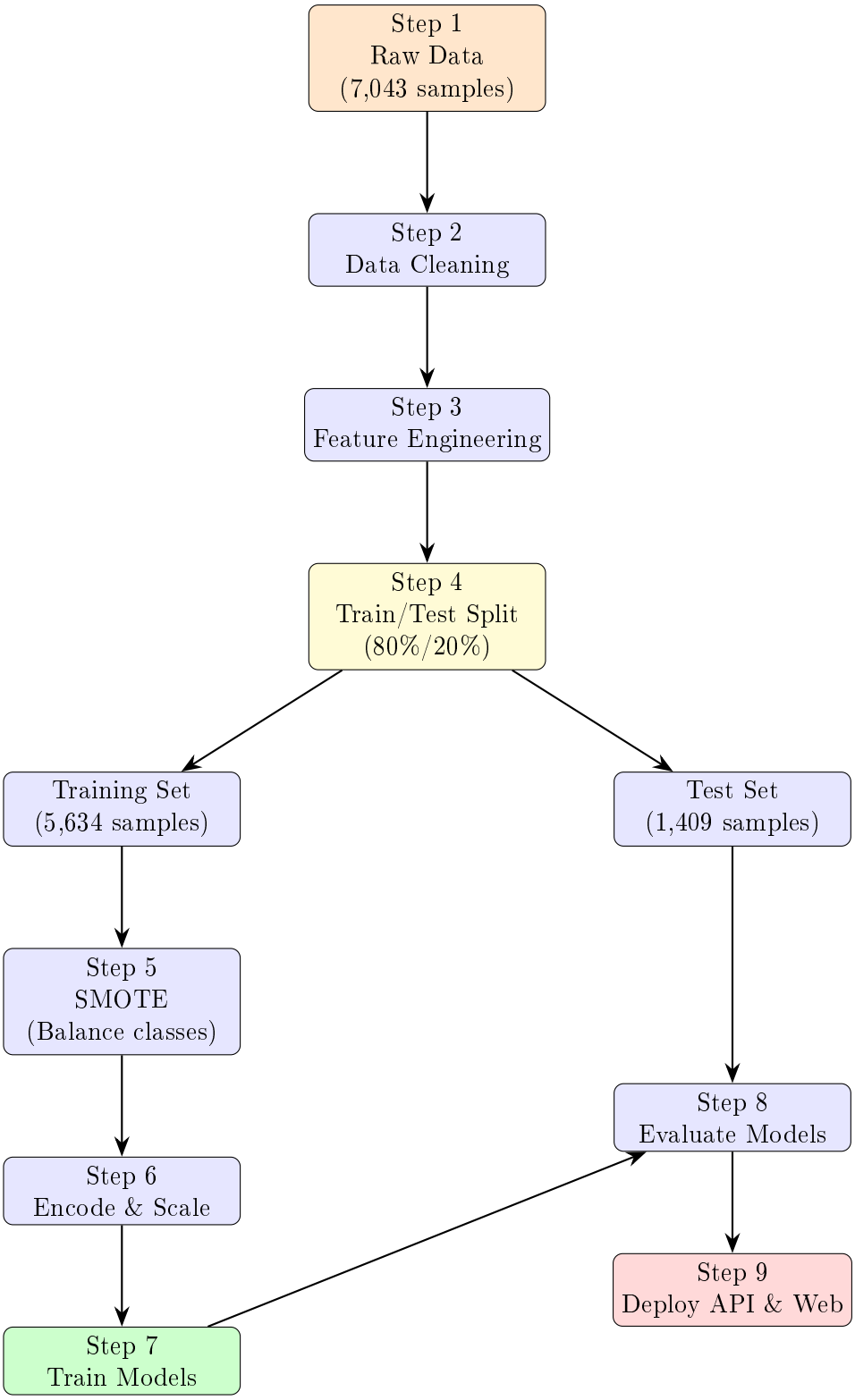

Figure 4: Our machine learning pipeline has 9 main steps. Notice that SMOTE is only applied to training data, not test data.

3.1 Step 1-2: Data Cleaning

Problem found: The TotalCharges column had 11 empty values (new customers who haven't been charged yet).

Solution: We filled these with 0, which makes sense because new customers have zero

charges.

3.2 Step 3: Feature Engineering

We created two new features to help our model:

Feature 1: Tenure Group

Instead of using the exact number of months, we grouped customers by their tenure phase:

Group	Tenure	Churn Rate
New Customer	0-12 months	47% (highest!)
Growing	12-24 months	28%
Established	24-48 months	19%
Loyal	48-72 months	13%
Very Loyal	72+ months	7% (lowest)

Table 1: Customers follow a predictable loyalty pattern.

Feature 2: Number of Services

We counted how many additional services each customer uses (security, backup, streaming, etc.). More services = more "locked in" = less likely to leave.

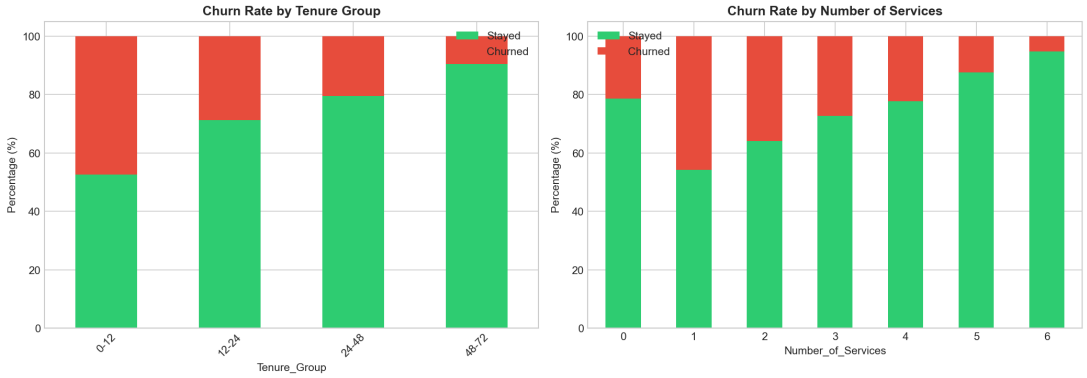

Figure 5: Both new features show clear patterns related to churn.

3.3 Step 4: Train/Test Split

We split the data into two parts:

- **Training set (80%):** 5,634 customers for teaching the model
- **Test set (20%):** 1,409 customers for honest evaluation

Critical Rule

We split the data **before** applying SMOTE. If we SMOTE first, fake data can leak into the test set and give us inflated (wrong) results!

3.4 Step 5: SMOTE (Handling Class Imbalance)

Problem: Only 26.5% of training data are churners. The model might ignore them.

Solution: SMOTE creates synthetic (artificial) examples of churners to balance the classes.

How SMOTE works:

1. Pick a churning customer
2. Find their nearest neighbors (similar churners)
3. Create new points between them

Mathematically: $x_{new} = x_i + \lambda \times (x_j - x_i)$ where λ is random between 0 and 1.
Result: Training set goes from 1,495 to 4,139 churners (now balanced 50:50).

3.5 Step 6: Encoding and Scaling

Encoding: Convert text to numbers so the model can understand.

- **Binary features** (Male/Female, Yes/No): Convert to 0/1
- **Multi-category features** (Contract type): Create separate 0/1 columns for each option

Scaling: Normalize numbers to similar ranges using the z-score formula:

$$z = \frac{x - \mu}{\sigma}$$

This makes features with large values (like TotalCharges) comparable to small ones (like tenure).

4 Evaluation Metrics Explained

For churn prediction, we use several metrics from the confusion matrix:

	Predicted: Stay	**Predicted: Leave**
Actually Stayed	True Negative (TN)	False Positive (FP)
Actually Left	False Negative (FN)	True Positive (TP)

Table 2: Confusion Matrix Layout

Key Formulas:

$$\text{Accuracy} = \frac{TP + TN}{\text{Total}} \quad \text{(Overall correctness—misleading for imbalanced data!)} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad \text{(Of those we flagged, how many were right?)} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad \text{(Of actual churners, how many did we catch?)} \tag{3}$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad \text{(Balance of both)} \tag{4}$$

For churn prediction, Recall is most important because missing a churner (False Negative) means losing a customer forever.

5 Model Training and Results

5.1 Models We Tested

1. **Logistic Regression:** Simple, interpretable baseline

2. **Random Forest:** Ensemble of 100 decision trees
3. **XGBoost:** Advanced gradient boosting algorithm
4. **Neural Network:** Deep learning with 2 hidden layers (16 and 8 neurons)

5.2 Results Comparison

Table 3: Model Performance on Test Set (Honest Evaluation)

Model	Accuracy	Precision	Recall	F1-Score	AUC
Neural Network	77.9%	56.8%	69.0%	**62.3%**	84.1%
Logistic Regression	74.3%	51.0%	**80.0%**	62.3%	84.1%
Random Forest	76.8%	55.2%	66.8%	60.5%	84.0%
XGBoost	78.1%	58.5%	60.7%	59.6%	83.4%

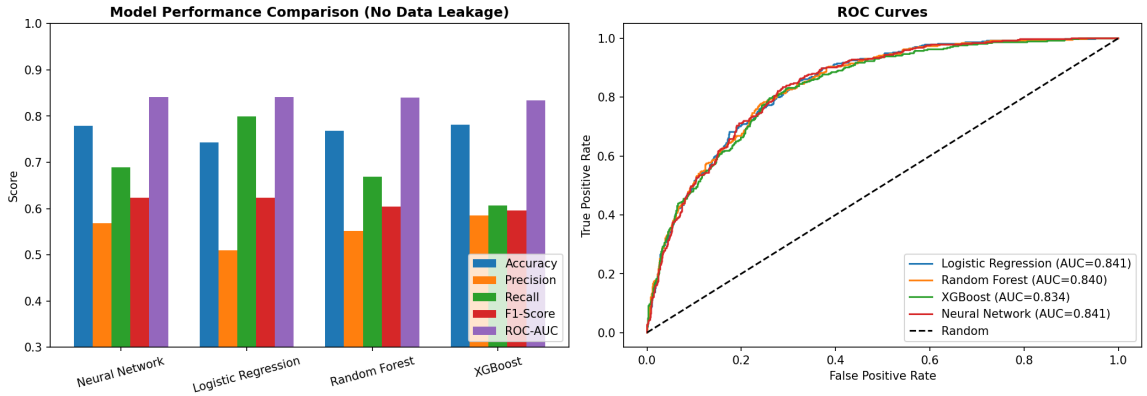

Figure 6: Left: Metrics comparison. Right: ROC curves showing all models have similar discriminative ability.

5.3 Confusion Matrices

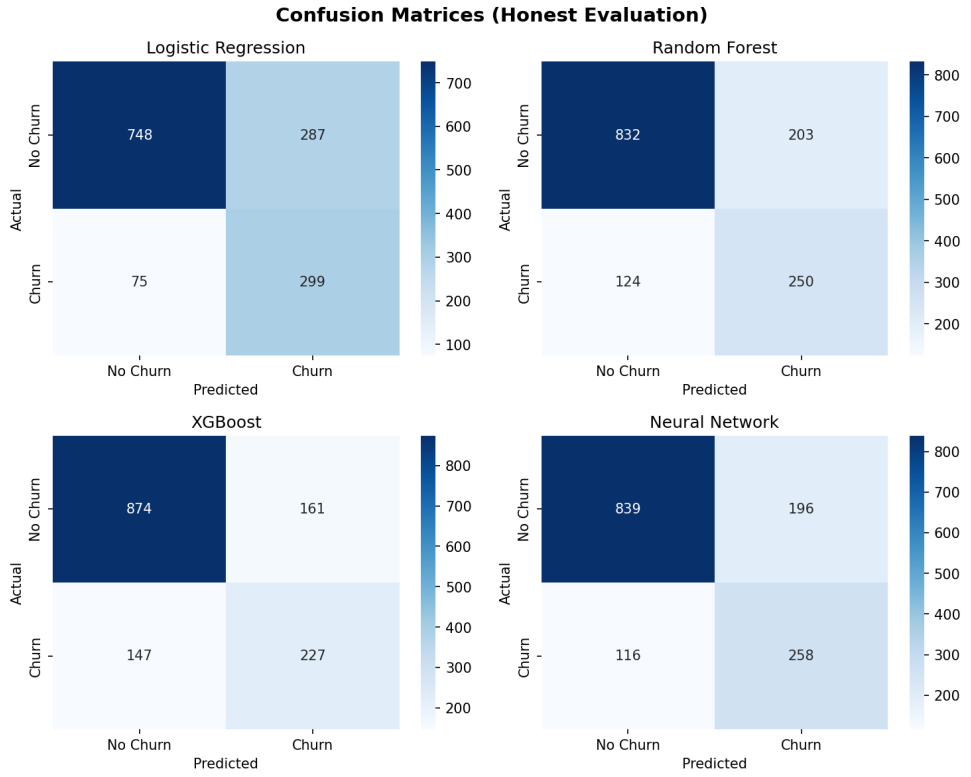

Figure 7: Confusion matrices for all four models.

5.4 Which Model is Best?

- **Best F1-Score:** Neural Network (62.3%)
- **Best Recall:** Logistic Regression (80%)—catches 80% of churners!
- **Best Precision:** XGBoost (58.5%)—fewest false alarms

Recommendation: Use Logistic Regression if you want to catch as many churners as possible, even at the cost of some false alarms.

5.5 What Features Matter Most?

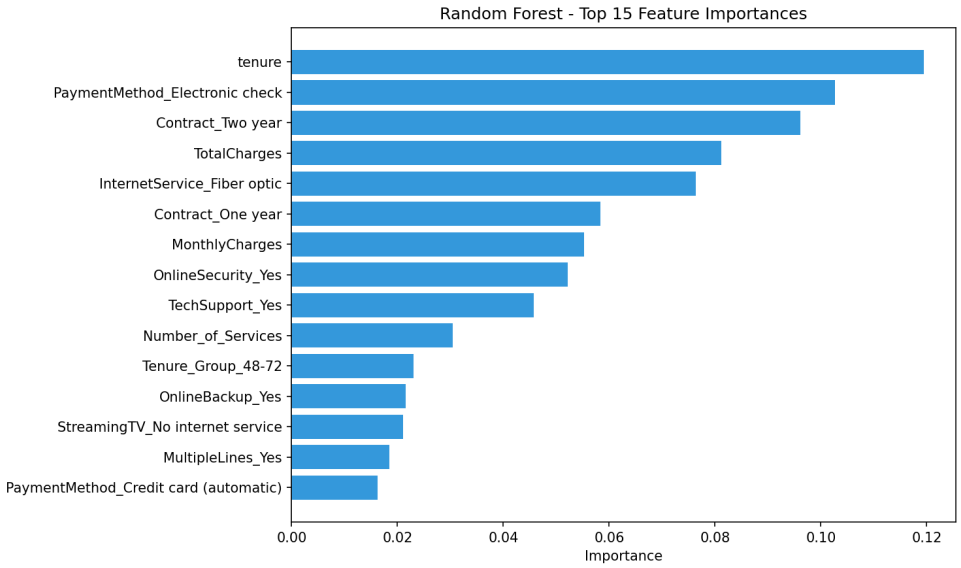

Figure 8: Top features according to Random Forest.

The most important predictors of churn are:

1. Total charges and monthly charges
2. Tenure (how long they've been a customer)
3. Contract type
4. Number of services

6 Deployment

6.1 REST API (FastAPI)

We created a web service that accepts customer data and returns predictions.

How to run:

1. Open terminal in project folder
2. Activate environment: `.\venv\Scripts\activate`
3. Start server: `uvicorn app.api:app -port 8000`
4. Open browser: `http://localhost:8000/docs`

6.2 Web Interface (Streamlit)

For non-technical users, we built an interactive web app.

How to run:

1. Start the API first (see above)
2. In a new terminal, run: `streamlit run app/streamlit_app.py`
3. Open browser: `http://localhost:8501`

7 Lessons Learned: Data Leakage

What went wrong initially:

We accidentally applied SMOTE before splitting the data. This caused synthetic (fake) churners to appear in our test set.

The evidence:

- Expected test set churn rate: 27% (natural)
- Actual test set churn rate: 50% (artificial!)

Impact:

- Before fix: F1-Score = 85% (fake, too good to be true)
- After fix: F1-Score = 62% (honest, realistic)

Lesson: Always split data before any data augmentation!

8 Business Recommendations

Based on our analysis, here are actionable recommendations:

Risk Factor	Churn Rate	Recommended Action
Month-to-month contract	43%	Offer discounts for 1-year upgrade
New customer (0-12 months)	47%	Special onboarding program
Electronic check payment	45%	Incentive for auto-pay enrollment
Few services subscribed	35%	Bundle package promotions
Fiber optic internet	42%	Check service quality issues

Table 4: Targeted retention strategies for high-risk segments.

9 Conclusion

9.1 What We Achieved

1. Built a complete ML pipeline from data to deployment
2. Trained 4 models including deep learning
3. Identified and fixed a critical data leakage issue
4. Created API and web interface for real-world use

9.2 Future Improvements

1. **Lower the prediction threshold** to catch more churners
2. **Tune hyperparameters** for better performance
3. **Add model explainability** using SHAP values
4. **Monitor performance** over time for drift