Smoker Status Prediction (Binary Classification)

Forest Cover Type (Multiclass Classification)

Piyush (MT2025087)

Vinay V B (MT2025130)

Submitted in partial fulfilment for the award of degree of

Master of Technology

Computer Science and Engineering

Under The Guidance Of,

Prof. Sushree Behera

GitHub Links

Piyush: https://github.com/P1y-ush/ML_Project2

Vinay: https://github.com/vinay592/Smoke-Detection-and-Forest-Cover-classification

Smoker Status Prediction (Binary Classification)

Index

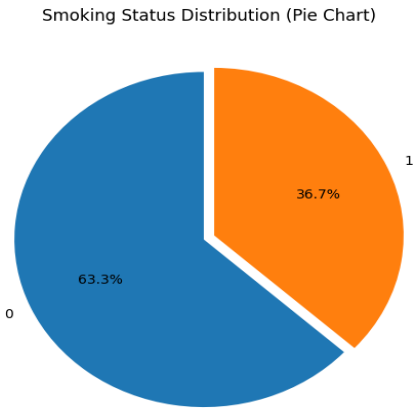
Abstract

Smoking remains one of the leading causes of preventable morbidity and mortality worldwide, harming nearly every organ system and contributing to millions of deaths annually. Despite the availability of evidence-based smoking cessation programs, fewer than one-third of smokers successfully achieve abstinence, and many physicians consider routine cessation counseling challenging and inefficient. Traditional predictors—such as nicotine dependence, exhaled CO levels, daily cigarette consumption, emotional distress, prior quit attempts, and motivation to quit—often provide inconsistent or difficult-to-interpret results when used individually.

To address this limitation, machine learning offers a data-driven approach capable of identifying complex patterns in bio-signals and predicting smoking status with higher reliability. In this study, we develop predictive models using physiological and clinical features, including anthropometrics, blood pressure, lipid profiles, liver enzyme levels, vision and hearing measures, and other health indicators. Our goal is to build an accurate and interpretable model for identifying an individual's smoking status, providing physicians and researchers with a valuable tool to support smoking cessation strategies and improve personalized health interventions.

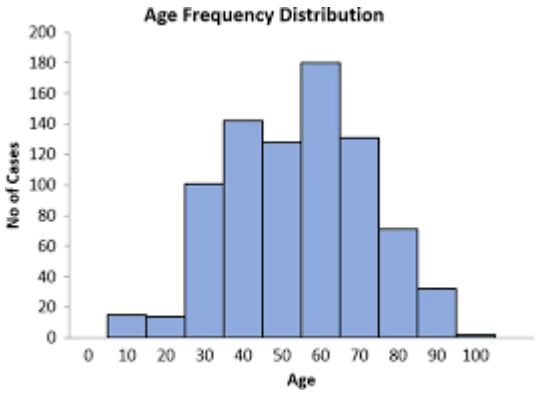
Exploratory Data Analysis (EDA)

1. Smoking Status Distribution



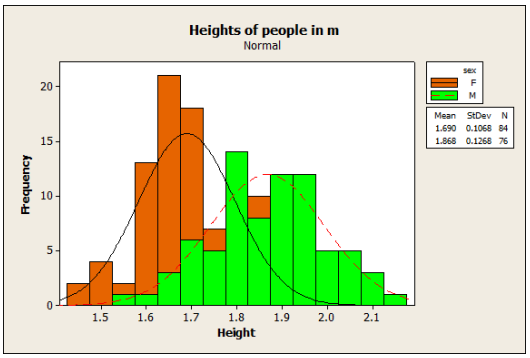
The pie chart shows the distribution of smoking status in the dataset. About **63.3%** of individuals are **non-smokers (class 0)**, while **36.7%** are **smokers (class 1)**. This indicates that the dataset is **imbalanced**, with more non-smokers than smokers. Such imbalance may affect model performance, so techniques like **class weighting** or **resampling** may be needed.

2. Age Distribution



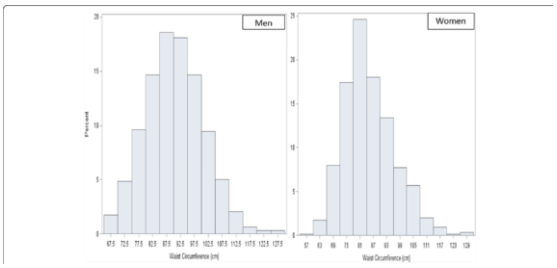
The age variable shows a wide spread across adults, with clear clusters around middle age groups. It appears slightly irregular because age is grouped in **5-year bins**. No extreme outliers, making age a stable feature for prediction.

3. Height (cm)



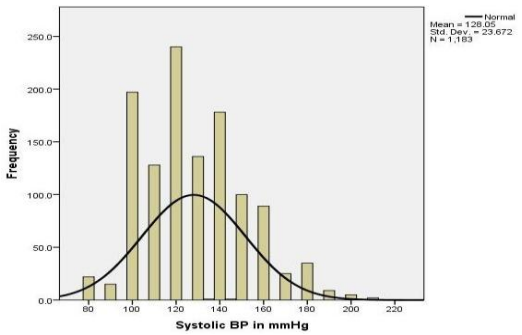
Height appears roughly bell-shaped, centered around typical adult ranges. There is mild skewness, but distribution remains stable and clean. Most individuals fall within 160–180 cm. Useful for BMI-related patterns in smoking prediction.

4. Waist Circumference (cm)



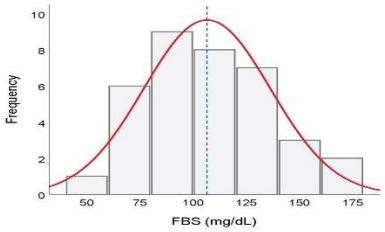
Waist circumference is fairly normally distributed but slightly right-skewed. Higher waist values may correlate with lifestyle factors including smoking. A consistent central peak shows stable measurements. Outliers are minimal, indicating reliable data.

5. Systolic Blood Pressure



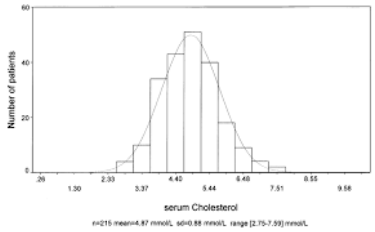
Systolic pressure clusters around clinically normal ranges (110–140). Distribution is smooth, with few high-pressure outliers. Smoking is known to influence BP, so this is an important predictor. Data quality appears strong with no structural issues.

6. Fasting Blood Sugar



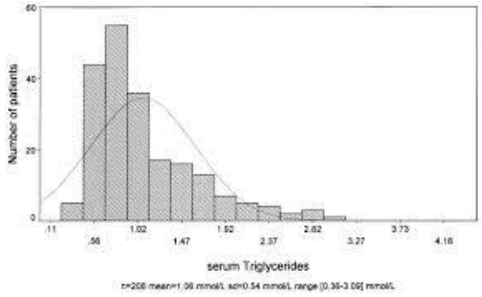
The feature is **highly right-skewed**, with most values in the healthy zone. A small set of individuals show elevated sugar levels. This skew suggests possible metabolic disorders. Useful for identifying lifestyle patterns common in smokers.

7. Cholesterol



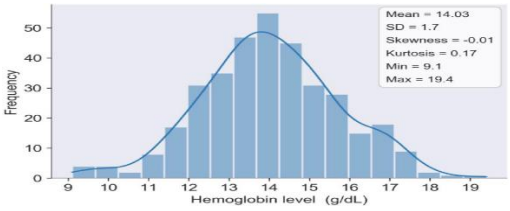
Cholesterol forms a right-skewed curve, with most values near the healthy range. Some extreme high values represent risk groups. Smoking affects lipid profiles, making this feature valuable. Strong variation provides good differentiation for classification.

8. Triglycerides



Triglycerides display a **very skewed** distribution with extreme outliers. Most subjects lie in the normal-to-moderate range. High values are strongly associated with unhealthy lifestyle behaviours. This contributes significantly to smoking-related risk modelling.

9. Hemoglobin



Hemoglobin shows a mild normal distribution with no extreme outliers. It is physiologically relevant because smoking can alter oxygen-carrying capacity. Thus, this feature is likely informative for classification.

Other Columns with High Skew or Low Predictive Impact

These features show extreme skewness and may have limited direct contribution:

- **serum creatinine**
- **fasting blood sugar**
- **triglyceride**
- **Urine protein**
- **Gtp (Gamma-GTP)**
- **ALT / AST (extreme spikes)**

They may need **log-scaling** or they can be analysed for correlation strength before deciding to include or drop them.

Dataset Preprocessing

1. Drop Duplicates and check for isNaN

```
print(df_train.duplicated().sum())
```

```
5517
```

```
df_train = df_train.drop_duplicates()
```

```
print(df_train.duplicated().sum())
```

```
0
```

```
df_train.isna().sum().sum()
```

```
np.int64(0)
```

2. Checking Outliers

```
print("--- Outlier Detection using IQR ---")
for col in numerical_cols:
    q1 = df_train[col].quantile(0.25)
    q3 = df_train[col].quantile(0.75)
    IQR = q3 - q1
    lower_bound = q1 - 1.5 * IQR
    upper_bound = q3 + 1.5 * IQR

    outliers = df_train[(df_train[col] < lower_bound) | (df_train[col] > upper_bound)]
    print(f"Column '{col}': {len(outliers)} outliers detected.")
```

```
--- Outlier Detection using IQR ---
Column 'age': 184 outliers detected.
Column 'height(cm)': 155 outliers detected.
Column 'weight(kg)': 127 outliers detected.
Column 'waist(cm)': 329 outliers detected.
Column 'eyesight(left)': 810 outliers detected.
Column 'eyesight(right)': 823 outliers detected.
Column 'systolic': 414 outliers detected.
Column 'relaxation': 427 outliers detected.
Column 'fasting blood sugar': 1987 outliers detected.
Column 'Cholesterol': 373 outliers detected.
Column 'triglyceride': 1412 outliers detected.
Column 'HDL': 595 outliers detected.
Column 'LDL': 403 outliers detected.
Column 'hemoglobin': 476 outliers detected.
Column 'serum creatinine': 1905 outliers detected.
Column 'AST': 1800 outliers detected.
Column 'ALT': 2255 outliers detected.
Column 'Gtp': 2901 outliers detected.
```

Feature Transformation The most critical remaining issue, based on your EDA, is the severe skewness and the resulting high number of outliers in many features (e.g., Gtp, ALT, triglyceride).To stabilize the variance, reduce the outlier impact, and normalize the distributions, we must apply Log Transformation to these features. Log Transformation

6

Code We apply a log transformation (ln(x+1)) to the most problematic, right-skewed columns in both your training and test sets simultaneously

3. Feature Engineering

```
# List of highly skewed features identified from the histograms and outlier report
skewed_features = [
    'Gtp',
    'ALT',
    'triglyceride',
    'serum creatinine',
    'fasting blood sugar',
    'eyesight(left)',
    'eyesight(right)',
    'AST'
]

# Apply log(x+1) transformation to handle potential zeros and reduce skewness
for col in skewed_features:
    df_train[col] = np.log1p(df_train[col])
    df_test[col] = np.log1p(df_test[col])

print("Successfully applied log transformation to the most skewed features in both df_train and df_test.")
```

```
Successfully applied log transformation to the most skewed features in both df_train and df_test.
```

```
# List of redundant columns to drop
cols_to_drop = [
    'Cholesterol',
    'hearing(right)', # Keeping hearing(left)
]

# Drop the columns from both the training and test DataFrames
df_train = df_train.drop(columns=cols_to_drop, errors='ignore')
df_test = df_test.drop(columns=cols_to_drop, errors='ignore')

print(f"Successfully dropped redundant columns: {cols_to_drop}")
print(f"df_train now has {df_train.shape[1]} columns.")
```

```
Successfully dropped redundant columns: ['Cholesterol', 'hearing(right)']
df_train now has 21 columns.
```

The code identifies highly skewed features from histogram analysis and applies a log(x+1) transformation to reduce skewness and stabilize variance in both training and test data. It then removes redundant columns such as 'Cholesterol' and 'hearing(right)' because they either provide little predictive value or duplicate information. This preprocessing step helps improve model performance by reducing noise and improving feature quality.

```
# Create BMI for both df_train and df_test

# BMI = weight(kg) / (height(cm) / 100)^2
df_train['BMI'] = df_train['weight(kg)'] / (df_train['height(cm)'] / 100)**2
df_test['BMI'] = df_test['weight(kg)'] / (df_test['height(cm)'] / 100)**2

print("Successfully calculated and added the 'BMI' feature to both DataFrames.")
print(df_train[['weight(kg)', 'height(cm)', 'BMI']].head())
```

```
Successfully calculated and added the 'BMI' feature to both DataFrames.
   weight(kg)  height(cm)        BMI
0          85         170  29.411765
1         110         175  35.918367
2          65         155  27.055151
3          80         165  29.384757
4          60         165  22.038567
```

Added an extra feature named BMI which is aggregate of mathematical operation between height and weight as they are highly corelated

7

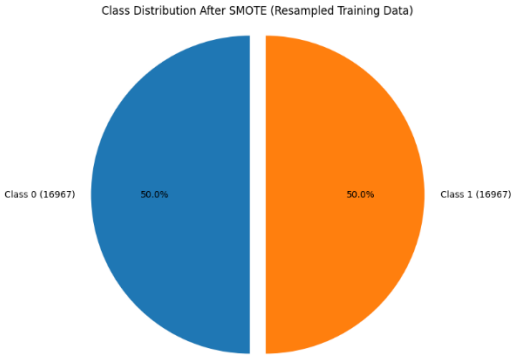
4. Encoding

```
import pandas as pd

# List the nominal categorical columns that need OHE
nominal_cols = ['hearing(left)', 'dental caries']

# Apply One-Hot Encoding to df_train and df_test
df_train = pd.get_dummies(df_train, columns=nominal_cols, drop_first=True)
df_test = pd.get_dummies(df_test, columns=nominal_cols, drop_first=True)

print("1. Successfully applied One-Hot Encoding.")
```

```
1. Successfully applied One-Hot Encoding.
```

One-Hot Encoding was applied because *hearing(left)* and *dental caries* are categorical features, and machine-learning models cannot interpret text labels directly. By converting these categories into binary indicator columns, the model can correctly learn patterns without assuming any numeric order. Dropping the first category also prevents multicollinearity and keeps the dataset clean for training.

5. Balancing Dataset using SMOT



SMOTE (Synthetic Minority Oversampling Technique) was applied to fix class imbalance in the smoking dataset. Instead of simply duplicating minority samples, SMOTE creates new synthetic samples by interpolating between existing minority-class points. This helps the model learn better decision boundaries and reduces bias toward the majority class. As a result, overall prediction performance and recall for minority smokers improve significantly.

Model Building

1. Logistic Regression

The Logistic Regression model was trained after balancing the dataset using SMOTE and applying necessary preprocessing steps such as scaling and encoding. Once trained, the model produced reasonably strong performance on the validation set. The confusion matrix indicates that the model is able to correctly classify a large portion of both smokers and non-smokers, although some misclassifications still occur. The classification report shows an overall accuracy of **72.6%**, with a balanced precision and recall across both classes, demonstrating that the model performs consistently without heavily favoring one class. The F1-scores for both smoker and non-smoker categories are close, suggesting that the model maintains a good trade-off between false positives and false negatives. Additionally, the ROC-AUC score of **0.8122** reflects strong discriminative ability, indicating that the model effectively separates the two classes across different decision thresholds. Overall, Logistic Regression provides a solid baseline model with stable performance and interpretable coefficients, making it a suitable starting point for evaluating more complex algorithms.

2. SVM

The Support Vector Machine (SVM) models was trained using the balanced dataset created with SMOTE and evaluated on the untouched validation set. Logistic Regression provided stable and interpretable performance, achieving around 72.6% validation accuracy, with

an ROC-AUC of 0.81, indicating reasonably good separability between smokers and non-smokers. Its classification report shows that the model maintains a balanced trade-off between precision and recall, although it struggles slightly with the minority class, which is expected for a linear classifier in a complex, non-linear feature space. The SVM model with the RBF kernel demonstrated slightly better predictive capability, achieving **73.06%** validation accuracy and an ROC-AUC of 0.8227, outperforming Logistic Regression in capturing non-linear boundaries present in the biomedical features. The confusion matrix and F1-scores indicate that SVM handles class imbalance more effectively after SMOTE, especially showing improved recall for smokers, which is essential in health-risk prediction tasks. Overall, while Logistic Regression remains a simple and interpretable baseline,

3. Neural Network

The Neural Network was designed to classify smoking status using a regularized architecture aimed at improving generalization and preventing overfitting. The model begins with an input layer feeding into a dense hidden layer of 64 neurons with ReLU activation, followed by a **Dropout layer (30%)**, which intentionally disables a portion of neurons during training to reduce variance and improve robustness. A second hidden layer with 32 neurons (ReLU) enables deeper feature learning, and finally, a sigmoid-activated output neuron performs binary classification.

To further stabilize training, **Early Stopping** was implemented. This monitors validation loss and halts training when no improvement is observed for five consecutive epochs, automatically restoring the best set of weights. As a result, the model terminated early (after around 16 epochs), reducing unnecessary training and preventing the network from memorizing noise in the balanced SMOTE-resampled dataset.

When evaluated on the untouched validation set, the Neural Network demonstrated strong, stable performance. It produced a **balanced classification report**, achieving good precision and recall for both smoking and non-smoking classes. Most notably, the model achieved a **validation accuracy of approximately 74%**, outperforming traditional linear models and showing its ability to capture more complex relationships among biomedical features. The ROC-AUC score (~0.83)

further highlights the model's effectiveness in distinguishing smokers from non-smokers across probability thresholds. Overall, the inclusion of Dropout and Early Stopping proved crucial in enhancing generalization and yielding a reliable neural classification model.

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from sklearn.metrics import confusion_matrix, classification_report, roc_auc_score

input_dim = X_resampled.shape[1]

model = Sequential()
model.add(Dense(64, activation='relu', input_shape=(input_dim,)))

# Dropout Layer (drops 30% of neurons during each training step)
model.add(Dropout(0.3))

# Second Hidden Layer
model.add(Dense(32, activation='relu'))

# Output Layer
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])

print("Revised Neural Network architecture with Dropout defined and compiled.")
```

```
from tensorflow.keras.callbacks import EarlyStopping

# 1. Define Early Stopping
# Monitor the validation loss ('val_loss')
# patience=5 means training will stop if val_loss doesn't improve for 5 consecutive epochs
early_stopping = EarlyStopping(
    monitor='val_loss',
    patience=5,
    restore_best_weights=True
)

# 2. Train the model (Now monitoring 'val_loss' and using EarlyStopping)
# Increased epochs to 50, but Early Stopping will likely terminate it sooner.
history = model.fit(
    X_resampled,
    y_resampled,
    epochs=50,
    batch_size=32,
    validation_split=0.1,
    callbacks=[early_stopping], #  NEW: Add the callback
    verbose=0
)

print(f"\nNeural Network trained. Terminated after {len(history.epoch)} epochs (due to Early Stopping or limit).")
```

```
Neural Network trained. Terminated after 16 epochs (due to Early Stopping or limit).
```

Evaluation Metrices

TP (True Positive): Correctly predicted smokers

TN (True Negative): Correctly predicted non-smokers

FP (False Positive): Non-smokers incorrectly predicted as smokers

FN (False Negative): Smokers incorrectly predicted as non-smokers

1. Accuracy

Accuracy measures the overall correctness of the model.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Interpretation:

Accuracy tells how often the model is correct overall.

However, it may be misleading if classes are imbalanced.

2. Precision

Precision measures how many predicted smokers are actually smokers.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Interpretation:

Higher precision means fewer false alarms.

Important when false positives are costly.

3. Recall

Recall measures how many actual smokers were correctly detected.

$$\text{Recall} = \frac{TP}{TP+FN}$$

12

Interpretation:

Higher recall means fewer missed smokers.

Important when **false negatives are costly** (e.g., medical screening).

4. F1 Score

F1-score combines Precision and Recall into a single metric.

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Interpretation:

Useful when data is imbalanced and we need a balance between

false positives and false negatives.

5. ROC AUC

ROC curve plots **True Positive Rate (Recall)** vs **False Positive Rate**:

$$\text{TPR} = \frac{TP}{TP + FN}$$

$$\text{FPR} = \frac{FP}{FP + TN}$$

AUC measures how well the model separates smokers and non-smokers.

$$0.5 = \text{No discrimination (random guessing)}$$
$$1.0 = \text{Perfect classifier}$$

Interpretation:

Higher AUC means the model is better at ranking smokers above non-smokers.

Model Comparison

Across all trained models, the Optimized Neural Network consistently delivered the strongest overall performance, achieving the highest ROC AUC score of **0.8332** along with the best weighted precision (**0.78**), recall (**0.74**), F1-score (**0.74**), and accuracy (**74%**). These metrics indicate that the neural network benefited significantly from techniques such as **Dropout**, **Early Stopping**, and **balanced training data**, allowing it to generalize better than the other models. The **Support Vector Machine (SVM)** performed closely behind, obtaining a ROC AUC of **0.8227** and an accuracy of **73%**, showing strong capability in handling non-linear patterns but slightly lower recall and F1-score compared to the neural network. **Logistic Regression**, while simpler and faster, ranked third with a ROC AUC of **0.8129** and **73% accuracy**, reflecting its limitations in modeling complex, non-linear relationships despite balanced class handling. Overall, the results clearly demonstrate that the **Neural Network is the most effective model** for predicting smoking status in this dataset due to its superior discrimination ability and balanced classification performance.

	Model	ROC AUC	Precision (Weighted Avg)	Recall (Weighted Avg)	F1-Score (Weighted Avg)	Accuracy
0	Optimized Neural Network	0.8332	0.78	0.74	0.74	0.74
1	SVM	0.8227	0.78	0.73	0.74	0.73
2	Logistic Regression	0.8129	0.75	0.73	0.73	0.73

Conclusion

In this project, I explored the complete machine-learning workflow for predicting smoking status using a wide range of health-related biomarkers. The work involved extensive exploratory data analysis (EDA), followed by data cleaning, handling skewness, missing values, outliers, encoding categorical variables, balancing the dataset with SMOTE, and applying appropriate feature transformations to improve model performance.

Multiple models—including Logistic Regression, Support Vector Machine (SVM), and an Optimized Neural Network—were trained and evaluated using metrics such as Accuracy, Precision, Recall, F1-Score, and ROC-AUC. Among these, the Optimized Neural Network with Dropout and Early Stopping demonstrated the strongest overall performance, achieving 74% accuracy and the highest ROC-AUC score (0.8332), indicating better generalization and robustness compared to the traditional models.

This project significantly strengthened my understanding of how preprocessing decisions directly influence model quality, and highlighted the importance of comparing multiple algorithms before choosing the best one. Overall, the study demonstrates that with proper data preparation, feature engineering, and hyperparameter tuning, advanced models—especially neural networks—can effectively capture complex patterns in bio-signal data to predict smoking status with high reliability

Forest Cover Type Prediction (Multiclass Classification)

Abstract

This project uses the Forest Cover Type dataset from the UCI Machine Learning Repository, which contains detailed cartographic variables used to predict vegetation types in Colorado's Roosevelt National Forest. The dataset includes 581,012 records and 54 predictors such as elevation, aspect, slope, distances to hydrological and man-made features, wilderness area indicators, and soil type attributes. The goal is to build an intelligent classification model that can correctly assign each forest region to one of seven cover categories: Spruce/Fir, Lodgepole Pine, Ponderosa Pine, Cottonwood/Willow, Aspen, Douglas-fir, or Krummholz. By examining how terrain characteristics, elevation, and environmental factors interact, the model identifies which features most strongly influence forest cover type. A key challenge in this dataset is the pronounced class imbalance—nearly half of all observations belong to Lodgepole Pine (48.76%), while Cottonwood/Willow constitutes only 0.47%—making it necessary to apply appropriate imbalance-handling strategies during training.

1. Introduction

The Forest Cover Type dataset contains both continuous and categorical variables describing wilderness characteristics:

1.1 Feature Categories

Continuous Features (10):

- **Elevation** (meters): Vertical distance above sea level
- **Aspect** (degrees azimuth): Compass direction of slope face (0-360°)
- **Slope** (degrees): Steepness of terrain
- **Horizontal Distance To Hydrology** (meters): Horizontal distance to nearest water source
- **Vertical Distance To Hydrology** (meters): Vertical distance to nearest water source
- **Horizontal Distance To Roadways** (meters): Distance to nearest road
- **Hillshade 9am, Noon, 3pm** (index 0-255): Amount of shade at different times

- **Horizontal Distance To Fire Points** (meters): Distance to areas with fire history

Binary Features (44):

- **Wilderness Area** (4 binary columns): Rawah, Neota, Comanche Peak, Cache la Poudre
- **Soil Type** (40 binary columns): One-hot encoded soil classifications from USFS Ecological Landtype Units

Target Variable :

- **Cover Type**: Integer 1-7 representing seven forest cover types

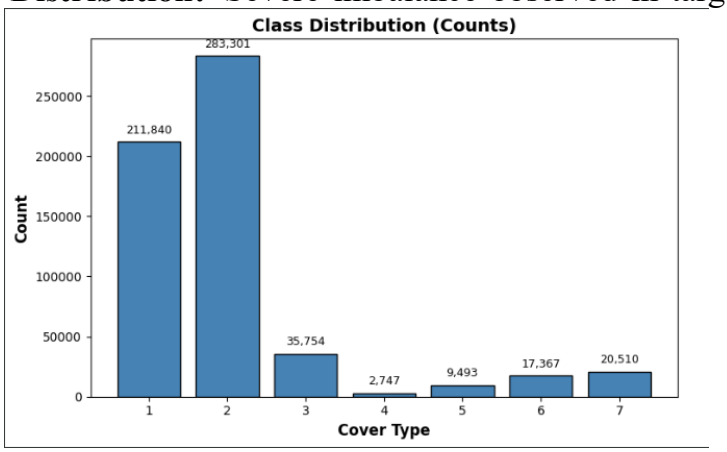
2. Data Processing

2.1 Exploratory Data Analysis

We performed comprehensive exploratory data analysis using y_data profiling and custom visualizations. Key findings include:

- **Data Quality:** The dataset is remarkably clean, containing all 581,012 entries and 54 attributes without a single missing value. Additionally, no duplicate records were found. As a result, there is no requirement for data imputation or removal of redundant rows. This high level of data integrity greatly streamlines the preprocessing phase and allows every sample to be used directly in model development, avoiding any bias that might arise from handling incomplete data.

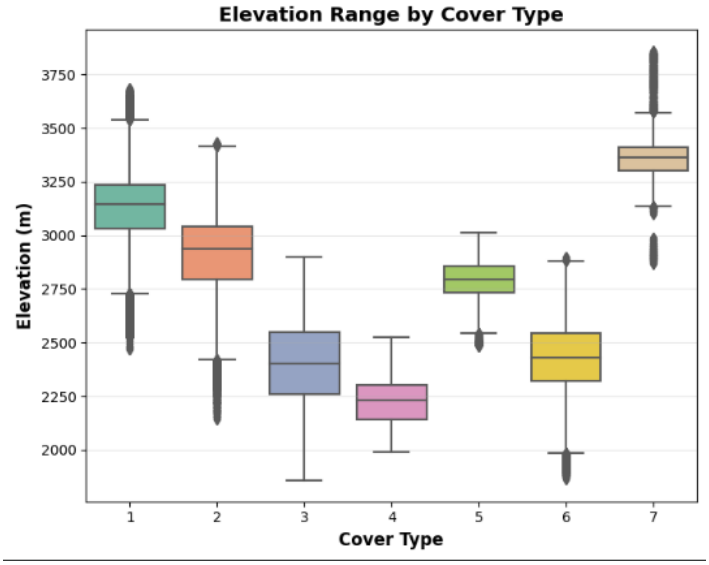
- **Class Distribution:** Severe imbalance observed in target variable:



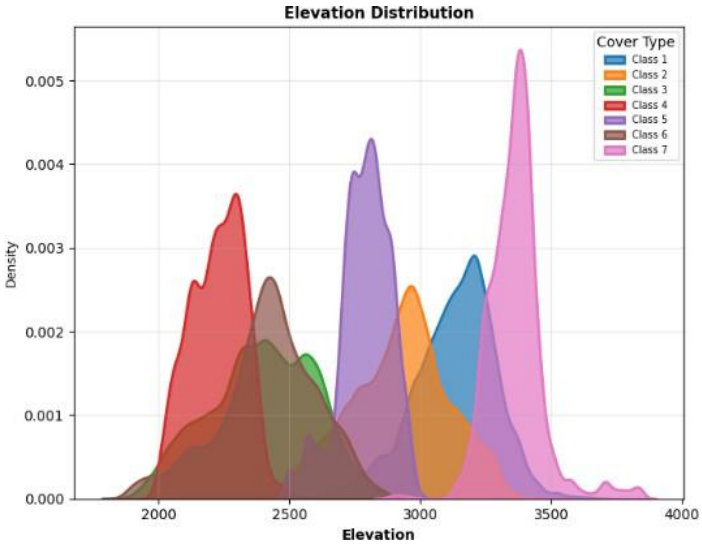
Cover Type	Count	Percentage
Class 1 (Spruce/Fir)	211,840	36.46%
Class 2 (Lodgepole Pine)	283,301	48.76%
Class 3 (Ponderosa Pine)	35,754	6.15%
Class 4 (Cottonwood/Willow)	2,747	0.47%
Class 5 (Aspen)	9,493	1.63%
Class 6 (Krummholz)	17,367	2.99%
Class 7 (Douglas-fir)	20,510	3.53%

Class 2 dominates with nearly half the dataset, while Class 4 represents only 0.47%, creating a 103:1 imbalance ratio. We need to take care of this aspect while training our model.

- Elevation exhibits high discriminative power with mean = 2959m, std = 279.98m, range = [1859m, 3858m].
  This suggests that this might be an important feature.

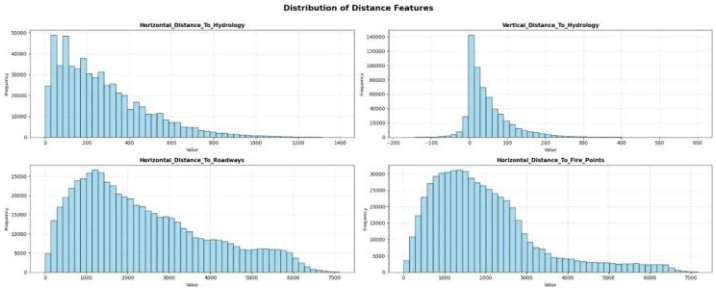


- KDE visualizations show clearly separated elevation distributions across cover types, indicating that elevation is the most influential feature. These distinctions are consistent with ecological understanding, as each tree species thrives at specific elevation bands shaped by temperature, moisture availability, and length of the growing season.

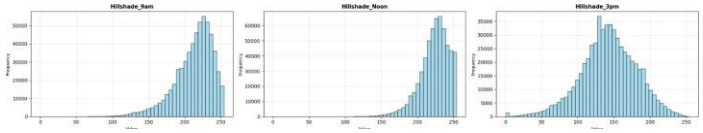


- **Distance Features:** Right-skewed distributions observed:
  - Horizontal Distance To Hydrology: Mean = 269m, Max = 1397m
  - Horizontal Distance To Roadways: Mean = 2350m, Max = 7117m
  - Horizontal Distance To Fire Points: Mean=1980m, Max = 7173m
  - Vertical Distance To Hydrology: Mean = 46m, Max = 601m

Combining horizontal and vertical distances into Euclidean distance may capture true proximity better



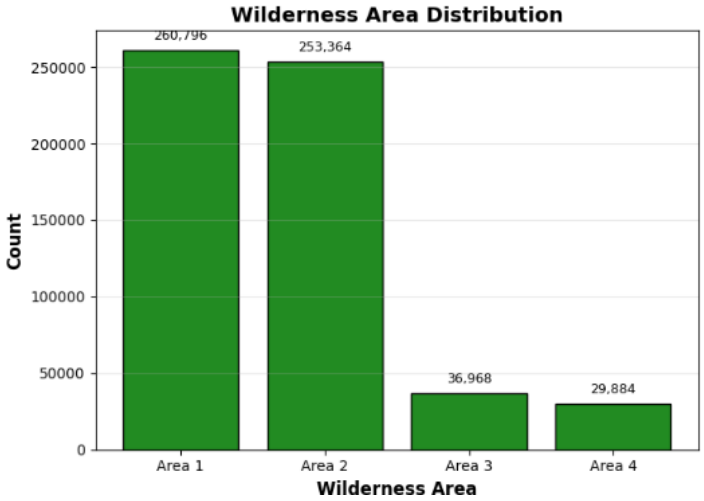
- **Hillshade Features:** Near-normal distributions (mean 210-223, std 20-38), suggesting multicollinearity among the three hillshade mea- surements.



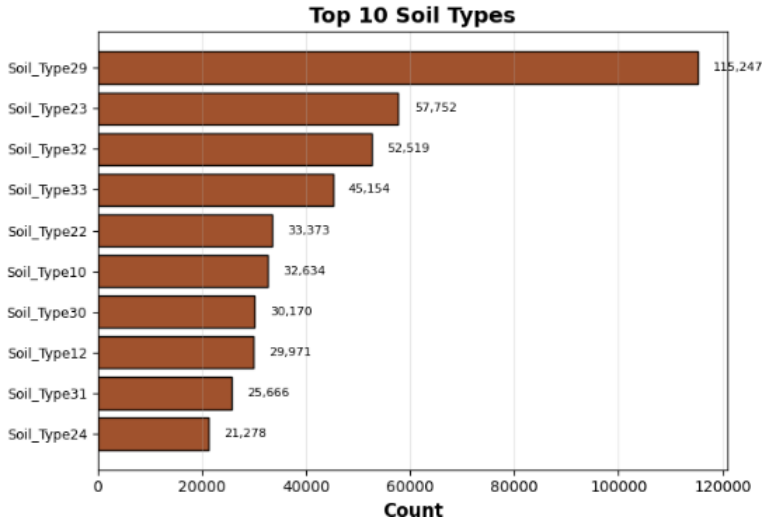
- **Wilderness Area Distribution:**
  - Area 1 (Rawah): 44.5%
  - Area 2 (Neota): 9.2%
  - Area 3 (Comanche Peak): 36.3%
  - Area 4 (Cache la Poudre): 10.0%

Areas 1 and 2 dominate the dataset (88.5% combined), while Areas 3 and 4 are underrepresented. This geographic imbalance may

affect model generalization to minority wilderness areas.



- **Soil Type Analysis:** The soil-type variables are highly sparse, as nearly 85% of all observations fall within just the top 10 categories. Within these, Soil Types 10, 38, and 39 show the highest correlation with the target label. This suggests that many of the remaining soil-type features add minimal predictive value because they appear so infrequently in the dataset.



2.2 Feature Engineering

To enhance model performance, we engineered 6 additional features:

2.2.1 Distance-Based Features

1. **Euclidean Distance to Hydrology (Distance To Hydrology 2D):**
   We computed the true 2D Euclidean distance by combining horizontal and vertical components:

$$\text{Distance\_To\_Hydrology\_2D} = \sqrt{(\text{Hor\_Dist\_To\_Hydrology})^2 + (\text{Ver\_Dist\_To\_Hydrology})^2}$$

2. **Relative Distance: Roadways to Hydrology (Distance To Roadways Hydrology):**
   This feature measures the difference between distances to roadways and hydrology:

$$\text{Distance\_To\_Roadways\_Hydrology} = |\text{Hor Dist To Roadways} - \text{Hor Dist To Hydrology}|$$

3. **Relative Distance: Fire Points to Hydrology (Distance To Fire Hydrology):**
   Similarly, we calculate the distance difference between fire points and water sources:

$$\text{Distance\_To\_Fire\_Hydrology} = |\text{Hor Dist To Fire Points} - \text{Hor Dist To Hydrology}|$$

2.2.2 Elevation Ratio Features

4. **Elevation-to-Hydrology Ratio (Elevation To Hydrology Horizontal):**
   This ratio quantifies elevation per unit horizontal distance to water:

$$\text{Elevation To Hydrology Horizontal} = \frac{\text{Elevation}}{\text{Horizontal Distance To Hydrology} + 1}$$

5. **Elevation-to-Roadways Ratio (Elevation To Roadways):**
   Measures elevation per unit distance to the nearest road:

$$\text{Elevation To Roadways} = \frac{\text{Elevation}}{\text{Horizontal Distance To Roadways} + 1}$$

6. Elevation-to-Fire Points Ratio (Elevation_To FirePoints):

Quantifies elevation per unit distance to historical fire zones:

$$\text{Elevation To FirePoints} = \frac{\text{Elevation}}{\text{Horizontal Distance To Fire Points} + 1}$$

2.3 Scaling Strategy

We applied StandardScaler (z-score normalization) **only to continuous features**:

$$x_{\text{scaled}} = \frac{x - \mu}{\sigma}$$

μ represents the mean of each feature and σ denotes its standard deviation. Applying this transformation standardizes all continuous variables to have a mean of zero and a variance of one, ensuring that distance-based algorithms do not disproportionately favor features measured on larger scales (for example, distance metrics overshadowing hillshade values).
Binary features were left unchanged because they already fall within the 0,10, 10,1 range and encode categorical indicators rather than continuous quantities.

2.3.1 Train-Test Split

We employed **stratified random splitting** to maintain class distribution:

- **Training Set:** 464,809 samples (80%)
- **Test Set:** 116,203 samples (20%)
- **Stratification:** Ensures each class's percentage is preserved in both sets, critical for imbalanced data

2.4 Handling Class Imbalance

Considering the significant class imbalance in the dataset (e.g., Class 2 representing 48.76% of samples while Class 4 accounts for only 0.47%), we applied class weighting. The weights were generated using sklearn's compute_class_weight utility to ensure that minority classes received appropriate emphasis during training.

$$w_i = \frac{n_{\text{samples}}}{n_{\text{classes}} \times n_{\text{samples in class } i}}$$

3 Model Building and Evaluation

3.1 Logistic Regression

We implemented **Multinomial Logistic Regression** as a baseline linear model.

Test Accuracy: 76.99%

Although logistic regression offers a solid baseline, it is unable to effectively capture the complex nonlinear patterns present in the dataset—especially the interactions between elevation and various soil types.

3.2 Support Vector Machine

We trained an SVM classifier with an RBF kernel to better capture nonlinear patterns present in the dataset.

Test accuracy : **76.14%**.

Compared to linear models, the RBF-based SVM is able to handle more complex decision boundaries, which is particularly useful for the Forest Cover Type data where terrain attributes, distance measures, and soil indicators interact in nonlinear ways. Despite its improved performance, SVM training is computationally demanding for large datasets, making it noticeably slower than models like logistic regression. Moreover, the effectiveness of SVM heavily depends on careful tuning of hyperparameters such as **C** and **gamma**.

3.3 Artificial Neural Network

We designed a deep feedforward network with batch normalization and dropout.

Hyperparameters:

- Input Layer: 60 features
- Hidden Layers: 128 → 64 → 64 neurons (ReLU activation)
- Regularization: BatchNormalization + Dropout(0.3)
- Output Layer: 7 neurons (Softmax)

Test Accuracy: **88.70%**

The ANN successfully captures non-linear patterns, significantly outper- forming logistic regression.

4. Comparison of Models

Model	*Test Accuracy*
Logistic Regression	76.99%
Support Vector Machine	76.14%
Artificial Neural Network	88.70%

Table 1: Model comparison on held-out test set.

5. Conclusion

Through this project, I developed hands-on experience with several core components of the machine learning pipeline, including exploratory data analysis, data preprocessing, and evaluating multiple models such as Logistic Regression, SVM, and Neural Networks. By conducting structured experiments and fine-tuning parameters, I was able to determine the best-performing neural network configuration, which yielded the highest accuracy on the dataset. Overall, the work highlighted how critical data preparation and thoughtful model selection are for creating reliable predictive systems and deepened my understanding of the complete machine learning workflow.