WEEK 1

# Introduction to Machine Learning

## Foundations, History, and the Economist's Perspective

ECO447 — Machine Learning for Economists

Assoc. Prof. Dr. Onur Polat
Hacettepe University, Institute of Informatics

# Today's Outline

1. What is Machine Learning?
2. ML vs Traditional Econometrics
3. Types of Learning: Supervised, Unsupervised, Reinforcement
4. The ML Pipeline
5. Bias-Variance Tradeoff
6. Model Evaluation Fundamentals
7. Python Ecosystem for ML

# 01

# What is Machine Learning?

KEY CONCEPT

# Machine Learning Defined

> *"A computer program is said to learn from experience E with respect to some task T and performance measure P, if its performance at T, as measured by P, improves with experience E." — Tom Mitchell (1997)*

Machine learning is the science of getting computers to act without being explicitly programmed. Instead of writing rules, we provide data and let algorithms discover patterns.

# Why ML for Economists?

- High-dimensional data: modern economic datasets have thousands of variables
- Prediction vs. inference: different goals require different tools
- Causal inference: ML methods enhance treatment effect estimation
- Text and unstructured data: NLP for policy analysis, sentiment in finance
- Nowcasting: real-time GDP, inflation, unemployment prediction
- Heterogeneous treatment effects: personalized policy evaluation
- Big data in finance: HFT, credit scoring, fraud detection

# A Brief History of ML

- 1950s: Perceptron (Rosenblatt), Turing Test
- 1960s-70s: Nearest neighbor, early neural nets, AI winter begins
- 1980s: Decision trees (CART, ID3), backpropagation revival
- 1990s: SVMs, Random Forests, boosting, kernel methods
- 2000s: Netflix Prize, ensemble methods dominate
- 2010s: Deep learning revolution, GPUs, ImageNet, AlphaGo
- 2020s: Foundation models, LLMs, causal ML, AI in economics

02

# ML vs Traditional Econometrics

# Econometrics vs Machine Learning

| Traditional Econometrics | Machine Learning |
| --- | --- |
| • Focus on causal inference & parameter estimation | • Focus on prediction accuracy |
| • Theory-driven model specification | • Data-driven model selection |
| • Hypothesis testing (p-values, t-stats) | • Cross-validation & holdout testing |
| • Unbiased estimators preferred | • Accepts bias for lower variance |
| • In-sample fit ($R^2$, F-test) | • Out-of-sample performance (MSE, accuracy) |
| • Asymptotic properties emphasized | • Finite-sample performance matters |
| • Small to moderate datasets | • Scales to massive datasets |

KEY CONCEPT

# Prediction vs. Inference

*Prediction: What will Y be given X? | Inference: How does X affect Y?*

Economists traditionally care about inference (β coefficients, causal effects). ML excels at prediction. Modern econometrics increasingly combines both: use ML for prediction tasks (e.g., nowcasting) and adapt ML for causal questions (e.g., causal forests, double ML).

# When to Use ML in Economics

- Prediction policy problems: who to target, where to allocate resources
- Variable selection: which regressors matter in high-dimensional settings
- Nonlinear relationships: when linear models are too restrictive
- Treatment effect heterogeneity: who benefits most from a policy
- Forecasting: GDP, inflation, asset prices, volatility
- Text-as-data: analyzing central bank communications, news sentiment
- Classification: credit default, fraud detection, recession prediction

# 03

# Types of Machine Learning

# Supervised Learning

- Given labeled data: $(X_1,Y_1)$, $(X_2,Y_2)$, ..., $(X_n,Y_n)$
- Goal: learn a mapping f: X → Y that generalizes to new data
- Regression: Y is continuous (house prices, GDP growth, returns)
- Classification: Y is categorical (default/no default, recession/expansion)
- Training: minimize a loss function L(Y, f(X)) on training data
- Validation: evaluate on unseen data to prevent overfitting
- Examples: linear regression, logistic regression, SVM, decision trees, neural nets

# Unsupervised Learning

- No labels: only features $X_1$, $X_2$, ..., $X_n$
- Goal: discover hidden structure, patterns, or groupings
- Clustering: group similar observations (K-means, hierarchical, DBSCAN)
- Dimensionality reduction: PCA, t-SNE, autoencoders
- Anomaly detection: identify unusual observations
- Association rules: find co-occurring patterns
- Economic applications: market segmentation, country classification, topic modeling

# Reinforcement Learning

- Agent learns through interaction with an environment
- Actions → Rewards/Penalties → Updated policy
- Goal: maximize cumulative reward over time
- Key concepts: states, actions, rewards, policy, value function
- Economic applications: dynamic pricing, portfolio optimization, auction design
- Less common in economics but growing: mechanism design, game theory
- Notable: AlphaGo, robotics, autonomous trading systems

# 04

# The ML Pipeline

# End-to-End ML Workflow

- 1. Problem Definition: frame the economic question as an ML task
- 2. Data Collection: gather and merge relevant datasets
- 3. Exploratory Data Analysis: understand distributions, correlations, patterns
- 4. Feature Engineering: create informative variables from raw data
- 5. Model Selection: choose appropriate algorithms
- 6. Training: fit the model on training data
- 7. Evaluation: assess performance on validation/test data
- 8. Interpretation: extract economic insights from the model

# Train / Validation / Test Split

- Training set (60-70%): model learns patterns from this data
- Validation set (15-20%): tune hyperparameters, select model
- Test set (15-20%): final unbiased performance estimate
- NEVER use test set for model selection — leads to overfitting
- Time series: use temporal splits (past → future), never random
- Cross-validation: K-fold for more robust estimates with limited data
- Stratified splitting: maintain class balance in classification tasks

05

# Bias-Variance Tradeoff

# The Bias-Variance Decomposition

$$E[(Y - \hat{f}(X))^2] = \text{Bias}^2(\hat{f}) + \text{Var}(\hat{f}) + \sigma^2$$

Expected prediction error = Bias² + Variance + Irreducible noise

$$\text{Bias}(\hat{f}) = E[\hat{f}(X)] - f(X)$$

Systematic error: how far the average prediction is from truth

$$\text{Var}(\hat{f}) = E[(\hat{f}(X) - E[\hat{f}(X)])^2]$$

Sensitivity to training data: how much $\hat{f}$ changes with new samples

$\sigma^2$ = irreducible error (noise in the data)

# Understanding the Tradeoff

- Simple models (e.g., linear): HIGH bias, LOW variance → underfitting
- Complex models (e.g., deep NN): LOW bias, HIGH variance → overfitting
- Optimal complexity: minimizes total prediction error
- Regularization: constrains model complexity to reduce variance
- More data generally helps reduce variance without increasing bias
- This is THE central concept in ML — every method addresses it
- In economics: OLS is high-bias/low-variance; kitchen-sink regression is the opposite

06

# Model Evaluation

# Regression Metrics

- MSE (Mean Squared Error): $(1/n)\ \Sigma(y_i - \hat{y}_i)^2$ — penalizes large errors
- RMSE (Root MSE): $\sqrt{}$MSE — in original units, more interpretable
- MAE (Mean Absolute Error): $(1/n)\ \Sigma|y_i - \hat{y}_i|$ — robust to outliers
- $R^2$ (Coefficient of Determination): 1 - SSR/SST — proportion of variance explained
- Adjusted $R^2$: penalizes for number of predictors
- MAPE: Mean Absolute Percentage Error — scale-independent
- Always report on TEST set, not training set

# Classification Metrics

- Accuracy: proportion of correct predictions — misleading with imbalanced data
- Precision: TP / (TP + FP) — of predicted positives, how many are correct?
- Recall (Sensitivity): TP / (TP + FN) — of actual positives, how many found?
- F1 Score: harmonic mean of precision and recall
- AUC-ROC: area under the receiver operating characteristic curve
- Confusion Matrix: complete picture of classification performance
- Economic context: cost of false positives vs. false negatives often asymmetric

07

# Python Ecosystem for ML

PYTHON

# Essential Python Libraries

```
# Core scientific computing
import numpy as np          # Arrays, linear algebra
import pandas as pd         # DataFrames, data manipulation
import matplotlib.pyplot as plt  # Visualization
import seaborn as sns        # Statistical visualization

# Machine Learning
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler

# Advanced ML
import xgboost as xgb       # Gradient boosting
import lightgbm as lgb      # Light gradient boosting
from scipy import stats     # Statistical functions
```

- scikit-learn: primary ML library with consistent API
- pandas: essential for economic/financial data wrangling
- statsmodels: bridge between econometrics and ML

PYTHON

# First ML Example: Predicting House Prices

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Load data
df = pd.read_csv("housing.csv")
X = df[["sqft", "bedrooms", "age", "distance_cbd"]]
y = df["price"]

# Split data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Train and evaluate
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(f"R²: {r2_score(y_test, y_pred):.4f}")
print(f"RMSE: {mean_squared_error(y_test, y_pred, squared=False):.2f}")
```

- Complete supervised learning pipeline in ~15 lines
- Same pattern applies to any scikit-learn model

# Cross-Validation in Practice

- K-Fold CV: split data into K folds, train on K-1, test on remaining fold
- Repeat K times, each fold serves as test set once
- Report mean and standard deviation of performance metric
- Common choices: K=5 or K=10
- Leave-One-Out CV (LOOCV): K=n, computationally expensive
- Time series: use TimeSeriesSplit — never shuffle temporal data
- Nested CV: inner loop for hyperparameter tuning, outer for evaluation

PYTHON

# Cross-Validation Example

```python
from sklearn.model_selection import cross_val_score, KFold

# 5-Fold Cross-Validation
kf = KFold(n_splits=5, shuffle=True, random_state=42)
scores = cross_val_score(
    model, X, y,
    cv=kf, scoring="neg_mean_squared_error"
)
rmse_scores = (-scores) ** 0.5
print(f"Mean RMSE: {rmse_scores.mean():.2f}")
print(f"Std RMSE:  {rmse_scores.std():.2f}")

# For time series - never shuffle!
from sklearn.model_selection import TimeSeriesSplit
tscv = TimeSeriesSplit(n_splits=5)
ts_scores = cross_val_score(model, X, y, cv=tscv,
                            scoring="neg_mean_squared_error")
```

- TimeSeriesSplit preserves temporal ordering
- neg_mean_squared_error because sklearn maximizes scores

SUMMARY

# Week 1 Key Takeaways

- ML learns from data rather than explicit rules — a paradigm shift for economists
- Prediction vs. inference: different goals, complementary approaches
- Three types: supervised (labeled), unsupervised (unlabeled), reinforcement (rewards)
- Bias-variance tradeoff is the central concept in model selection
- Always evaluate on out-of-sample data — in-sample fit can be deceptive
- Python (scikit-learn, pandas, numpy) provides a mature ML ecosystem
- Next week: Causality — where economics and ML perspectives diverge and converge