Classification of Film Reviews using Machine Learning

MINI PROJECT

AAKASH JAIN (120301005) & PUNIT CHHAJER (120301059)

What

- Given a movie review, analyze sentiment to predict the star rating.

- Without using NLP techniques, find best machine learning approach
- Evaluate performance of learning models with various parameters
- Create webapp to perform prediction

Why

- Quantify reviews that don't have ratings
- Analyze comments about movie trailers/previews to gauge public opinion

- NLP approaches often don't generalize while machine learning might
- Solution may be domain specific, hence needs parameter tuning

How

- Scikit-learn
  - MultinomialNB
  - LinearSVC
  - LinearSVR
  - LogisticRegression
  - LinearRegression
- IPython (Jupyter) for experiments
- Flask for webapp
- ACL IMDB Dataset

ACL IMDB dataset

- 50,000 reviews classified as pos/neg and split into test/train
- Directory structure is perfect for scikit-learn
- We reorganize into star categories (Filenames are <id>_<stars>.txt)
- Form aggregate dataset for custom splits

- 5 & 6 star reviews not available
- 1 & 10 star reviews 2x as many as other classes

A few terms

- Classification: From a set of distinct target classes, assign one to the input
- Regression: Given the input, predict the output as a continuous variable
- Accuracy: How many inputs are correctly classified
- Confusion matrix: Shows for each class, how many of its inputs were classified as which class. Can be more useful than accuracy.

Searching for the best models

- First as positive/negative sentiment, then as stars
- Experiment workflow
  1. Load training and testing sets
  2. Convert reviews to tf-idf vectors
  3. Grid-search chosen algorithm on all combinations of relevant params
  4. Test the optimal model and record metrics

Notebooks

Command line app

- Framework that uses scikit-learn to
  - Train classifiers and dump to disk (python pickle)
  - Classify a review using a dumped classifier
  - Various other utilities to deal with the dataset and dumps
  - Can be used as a library or standalone

Code walkthrough

Dumps

- Python allows serialization of objects
- Instead of training classifier before every prediction, load from disk
- Dumping process is very slow
- Dumps need large amounts of storage space
- Loading is slow but faster than training

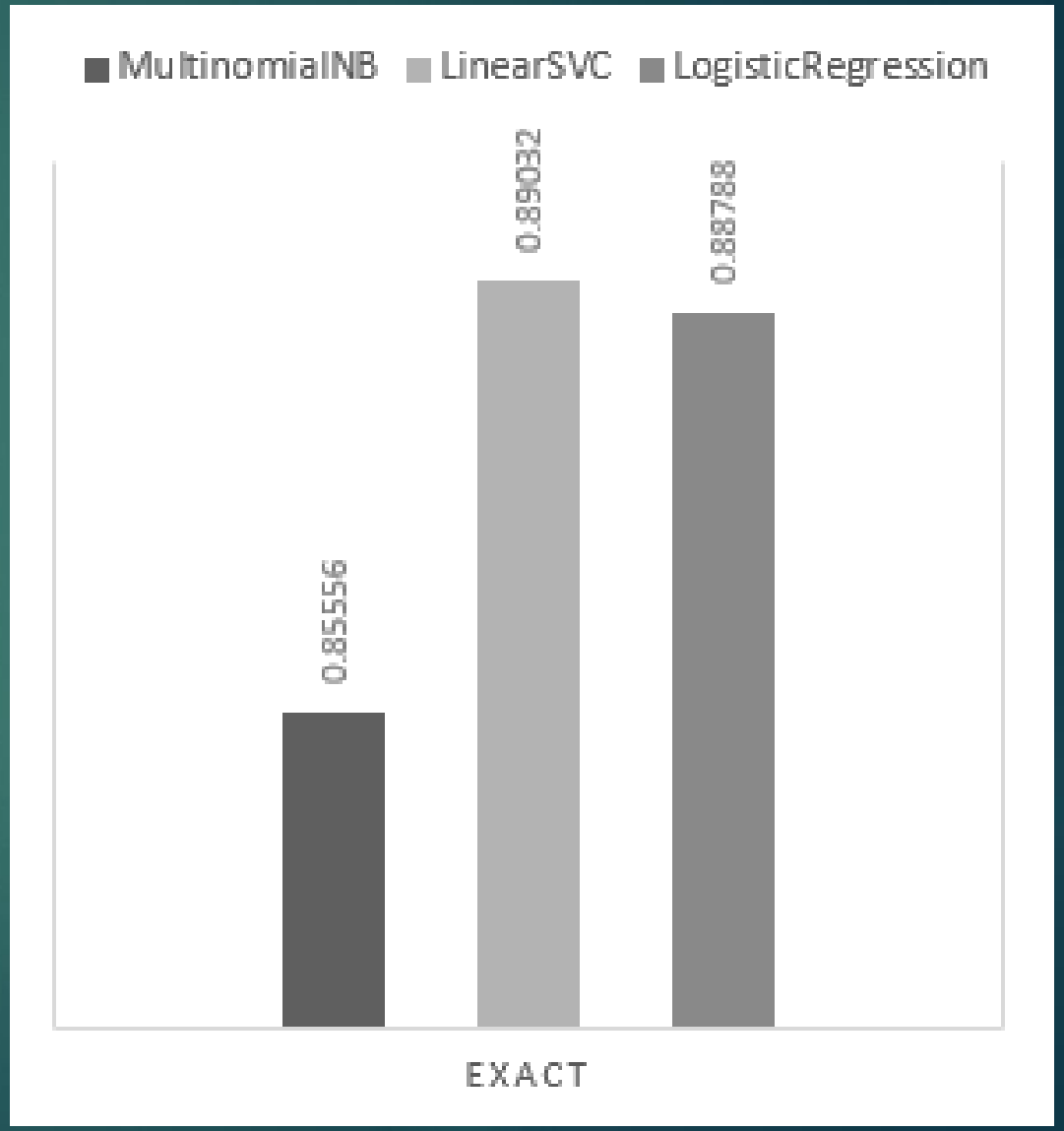
Webapp

- Loading dumped classifiers causes delay
- Solution: load once and provide an interface
- Models used
  - LinearRegression for star ratings
  - LinearSVC for positive/negative
- Flask for backend
- Jinja template for frontend

Webapp demo

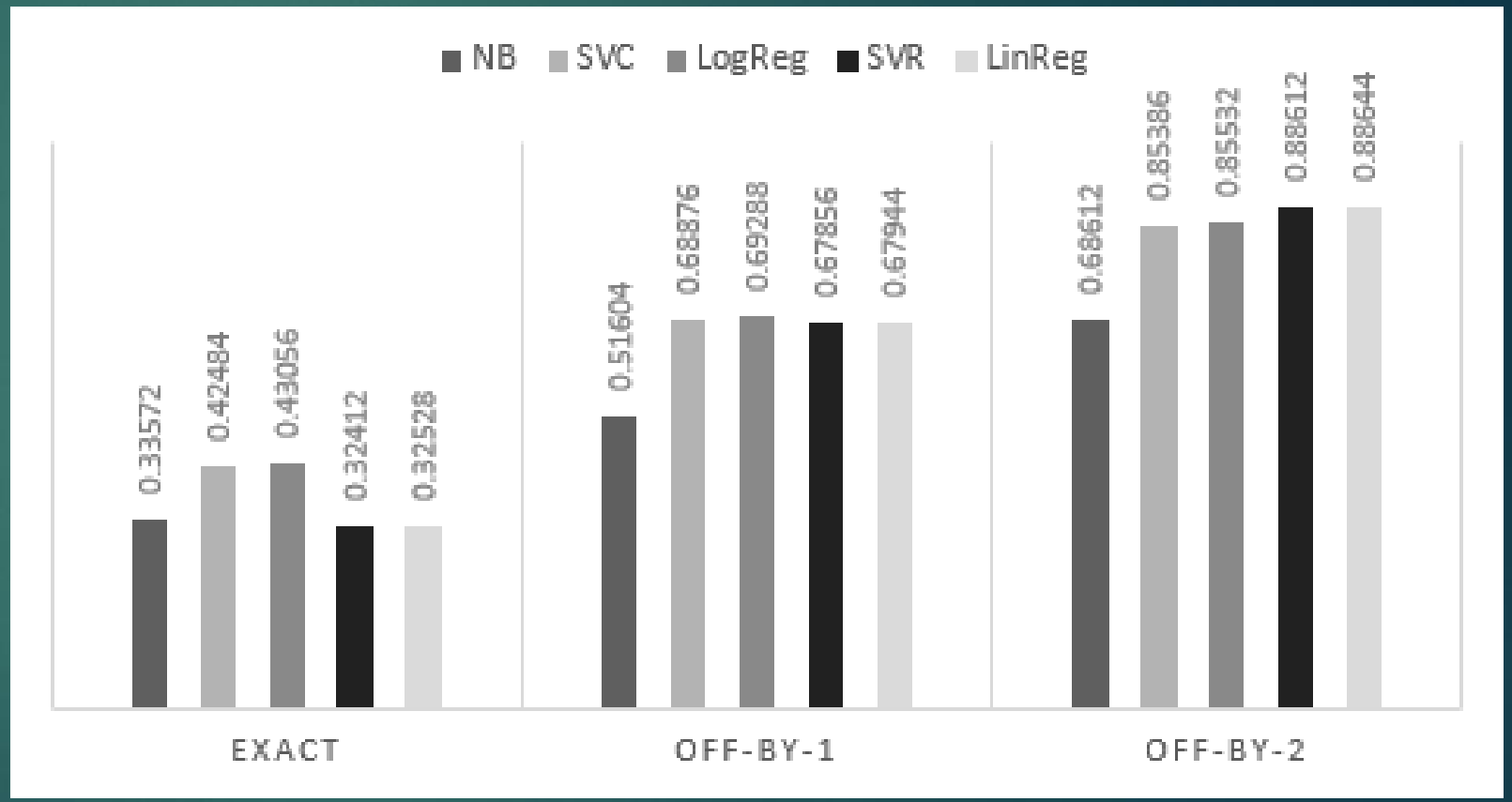
Results – Binary classification

- All 3 classifiers (MultinomialNB, LinearSVC, LogisticRegression) perform well
- LinearSVC best at 89% accuracy
- Satisfactory solution reached



Results – Star classification

- Regressors perform poorly with exact accuracy in mind
- Regressors perform acceptably if off-by-one errors are allowed (LinearRegression: 68%)
- Positive/negative distinction maintained
- But accuracy needs to be improved



Thank you