Variational Autoencoders

Introduction
- Autoencoders are a type of neural network architecture used to learn efficient representations of data. Variational autoencoders (VAEs) are a type of generative model that synthesizes new examples from learned representations.
- VAEs is a type of Bayesian latent variable model.
- VAE architecture (figure 1) consists of an encoder that encodes data into efficient embeddings and a decoder that translates the code back to the data representation. VAEs add stochastic noise to generate new samples.

Methods
- VAE implemented from the paper Tutorial on Variational Autoencoders (Doersch, 2016) from scratch using Python, numpy, and scipy.
- Built Python package using software development best principles including unit tests and version control.
- Benchmarked performance of Python implementation compared to TensorFlow implementation.

Results
- Compared to TensorFlow benchmark, the native Python implementation is much slower on the MNIST dataset. This is likely due to efficiency improvements within the TF backend and Python’s interpreted overhead.
- VAEs are know to produce blurry images. Our implementation does a poor job of learning from the data (figure 2) due to long training times.

Automatic Detection of Diabetic Retinopathy

Introduction
- Diabetic retinopathy is a disease affecting people with diabetes mellitus which can lead to blindness.
- Standard procedure has been to have trained ophthalmologists grade retinal fundus images (figure 1) on a 0 (no disease) to 4 (severe proliferative disease) scale.

Methods
- Dataset consists of over 100,000 retinal fundus photographs. Preprocessing includes scaling, cropping, and background normalization.
- Convolutional neural networks (CNNs) (figure 2) were trained on a training set with holdout sets for testing and validation. Models were written in TensorFlow and Keras.
- Overcame model overfitting issues with data augmentation. Used weighted loss function due to class imbalance.
- Several candidate models were trained, considering tradeoffs between performance and model training time.

Results
- Agreement to labels measured with quadratic weighted kappa. Accuracy of best performing model around 80% with a QWK of 0.6. This is comparable with human inter-rater agreement, with slightly lower sensitivity and specificity.

Predicting Insurance Claims

Introduction
- Dataset for insurance claim severity taken from the Allstate claims severity Kaggle competition. The objective is to predict loss (in USD) for each incident given.
- The data consists of over 130,000 rows and 130 anonymized features (14 continuous, 116 categorical).
- Predictions were evaluated using mean absolute error between predicted loss and actual loss.

Methods
- Gradient boosted trees (with xgboost) were chosen because they are flexible, have many parameters to tune, and performed well historically in similar Kaggle competitions.
- Exploratory analysis show strongly left-skewed distribution of losses (figure 1a). A Box-Cox transformation was performed on the target and other continuous variables.
- Categorical variables were transformed into one-hot encoded matrix representation.
- Instead of squared error loss, log cosh error (figure 1b) was used because it more closely approximates the mean absolute error loss while still being twice differentiable.
- Other methods for improvement include ensembling predictions from three models initialized with different random seeds as well as feature engineering of interactions.

Results
- Out of sample performance was monitored using five-fold cross-validation on a holdout set.
- Submissions are ranked on a public test set and a private test set. Submitted top two results indicated by CV.
- Placed first out of 34 submissions on the private leaderboard.

Project for STA 663 Statistical Computation.