Final Exam - Applied Machine Learning COMS W4995

Date: 05/01/17

Name:

UNI:

1 True/False (+2 for correct/ - 2pt for incorrect, +/- 0 for unanswered.)

	True	False
The decision boundary learned by a neural network with tanh activation function will be piecewise linear.		x
Comparing two models, the one with higher AUC will also have the higher accuracy.		x
PCA components are always orthogonal	x	
For the same number of components, PCA will always provide as least as good a reconstruction MSE than NMF.	x	
T-SNE can only find linear relationships in the data.		x
In imbalanced classification tasks, upsampling generally provides better models than downsampling.		x
Outlier detection with EllipticEnvelope assumes Gaussian distributed data.	x	
Short strings should always be treated as categorical variables.		x
Training a densely connected neural network on images ignores the neighborhood structure of pixels.	x	
Bag-of-word models using bigrams completely ignore the order of words in a sentence.		x

2 Multiple choice (21pt /1 for each correct answer)

Select all choices that apply.

2.1 Which of the following models can easily be used for outlier detection on continuous numeric data?

- ❏ Random Forests
- ✓ Gaussian Mixture Models
- ✓ Isolation Forests
- ❏ Logistic Regression
- ✓ PCA
- ❏ Latent Dirichlet Allocation

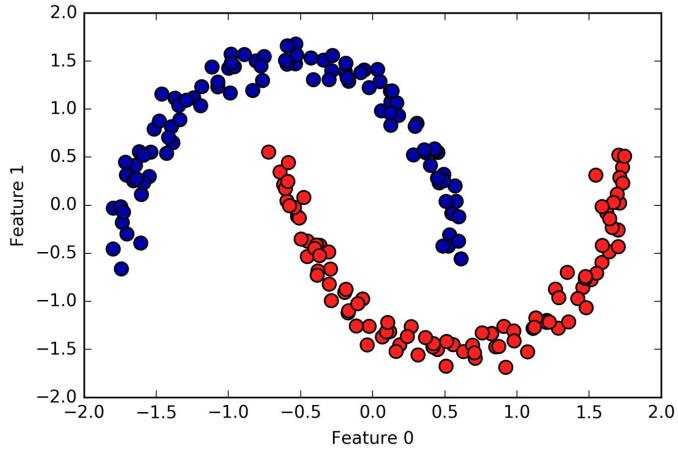
2.2 What are possible ways to speed up training a neural network.

- ✓ Using GPUs for the computations.
- ❏ Using drop-out.
- ✓ Using batch normalization.
- ✓ Initializing the network with weights trained on a similar task.
- ❏ Increasing the number of samples in each mini-batch.

2.3 Which of the following metrics (by themselves) are appropriate for model selection in an imbalanced binary classification task.

- ❏ Accuracy
- ✓ Area under the ROC curve
- ✓ F1 score
- ❏ Precision
- ✓ Average precision
- ❏ False positive Rate

2.4 Which of the following clustering algorithms could potentially discover this clustering?



- ❏ KMeans
- ✓ DBSCAN
- ❏ Agglomerative clustering without topology constraint.
- ✓ Agglomerative clustering with topology constraint.

3 Debugging (10pt each)

For each code snippet, find and explain all errors given the task. Assume all necessary imports have been made. There can be more than one error per task!

3.1 Task: Define a multi-layer perceptron using Keras for use on MNIST with drop-out regularization.

```
model = Sequential()
model.add(Dense(512))
model.add(Dropout(.5)
model.add(Dense(512))
model.add(Dropout(.5)

model.compile(“adam”, loss=”multiclass_crossentropy”,
              metrics=[“accuracy”])
```

Missing input shape. 3pts
Missing non-linearities. 3pts
Missing output layer. 4pts

Missing parenthesis: +1 bonus each
Loss should be “categorical_crossentropy” +1 bonus

3.2 Task: Implement an autoregressive model for a one-dimensional time-series using LinearRegression and predict for 5 steps. The time_series variable is a 1d array.

```
X = []
window_size = 5
y = time_series[:-window_size]
for i in range(len(y)):
     X.append(time_series[i: i + window_size])
lr = LinearRegression().fit(X, y)

prediction = []
for step in range(5):

prediction.append(lr.predict([prediction[-window_size:]])[0])
prediction = prediction[window_size:]
```

Y should not start at the same place as X (should be y[window_size:]) 3pts
Prediction needs to be initialized with the last steps of time_series: 4pts
prediction = time_series[-window_size:].tolist() 3pts

4 Coding (10 each)

Assume all necessary imports have been made.

4.1 Apply Latent Dirichlet Allocation with 20 topics to a text corpus given as `text_data`. For each topic, print the 10 most important words.

```
# it would be a good idea to add options like stopwords and min_df here
cv = CountVectorizer() 1 pt
X = cv.fit_transform(text_data) 2pts (fit + transform)
lda = LatentDirichletAllocation(n_components=20) 1pt (init,
parameter)
lda.fit(X)  1pt
feature_names = np.array(lda.get_feature_names()) 1pt
for comp in lda.components_: 1pt
    inds = np.argsort(comp)[::-1][:10] 2pt (sorting + reverting)
    print(feature_names[inds]) 1pt
```

4.2 Apply PCA to a dataset given in `X` (include the necessary preprocessing). Plot the importance of each of the components (scree plot) for all components, and do a scatter plot of the first two principal components, colored by labels given by `y`.

```
X_scaled = scale(X)
pca = PCA()
# alternatively:
X_scaled = StandardScaler().fit_transform(X)
pca = PCA()
# alternatively:
pca = make_pipeline(StandardScaler(), PCA())
# Either variant: 4pts (instantiate + fit + transform +
instantiate)
# -1 if n_components specified (makes scree-plot useless)

X_pca = pca.fit_transform(X_scaled)  2pts (fit + transform)
plt.plot(pca.explained_variance_ratio_) 2pts
plt.scatter(X_pca[:, 0]. X_pca[:, 1]) 2pts
```

5 Concepts (5pt each)

Answer each question with a short (2-5 sentences) explanation.

5.1 When would you want to use “Easy Ensembles” (bagging with resampling), and how do they work?

- They can improve performance on highly imbalanced datasets.
- Each estimator in the ensemble is trained on a different balanced subsampling of the data.

Both present: full points, one present 3pts

Additional explanation:
That allows using many samples of the majority class while still training each model on a balanced dataset.

5.2 Compute the number of parameters in a convolutional neural network with 16x16x1 input, followed by two 3x3 convolution layers with 4 maps each, followed by a 2x2 max pooling layer followed by an output layer with two units (don’t forget biases). You can just write out the multiplications and additions for each layer, you don’t need to compute the additions and multiplications.
1 * 3 * 3 * 4 + 4 filter + bias for first conv layer
+ 4 * 3 *3 * 4 + 4 filter + bias for second conv layer
+ 6 * 6 * 4 * 2 + 2 dense layer weights + biases. Resolution after first layer is 14, second 12, pooling 6 -> 6 * 6 * 4 hidden units.
# 1 for getting first two layer sizes right, 1 for filter sizes 1 for bias 2 for last layer size

5.3 Given a function “vec” that computes the word2vec representation of a word, what do you have to compute to answer the question “python is to snake as matlab is to X”

Find the word for which the embedding is closest to vec(“matlab”) + vec(“snake”) - vec(“python”)
3pts for formula, 2pts for “find vector closest to”

5.4 Give two strategies to remove the trend from a time-series.
Global model (linear etc) and subtract, or diff. (2.5 each)