Final Exam - Applied Machine Learning COMS W4995

Date: 05/01/17

Name:

UNI:

For all choice boxes, please fill in the box you want to choose like this: ■
Otherwise your answer can not be graded.

1 True/False (+2pt each)

	True	False
The decision boundary learned by a neural network with tanh activation function will be piecewise linear.	☐	■
Adding drop-out to a neural network will increase the number of parameters.	☐	■
PCA components are always orthogonal.	■	☐
For the same number of components, PCA will always provide at least as good a reconstruction MSE as NMF.	■	☐
T-SNE can only find linear relationships in the data.	☐	■
NMF can not be applied to sparse data.	☐	■
Outlier detection with EllipticEnvelope assumes Gaussian distributed data.	■	☐
DBSCAN does not allow directly specifying the desired number of clusters.	■	☐
In Latent Dirichlet Allocation, each word can occur in multiple topics	■	☐
Bag-of-word models using bigrams completely ignore the order of words in a sentence.	☐	■

2 Multiple choice (20pt)

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
- ☐ Latent Dirichlet Allocation

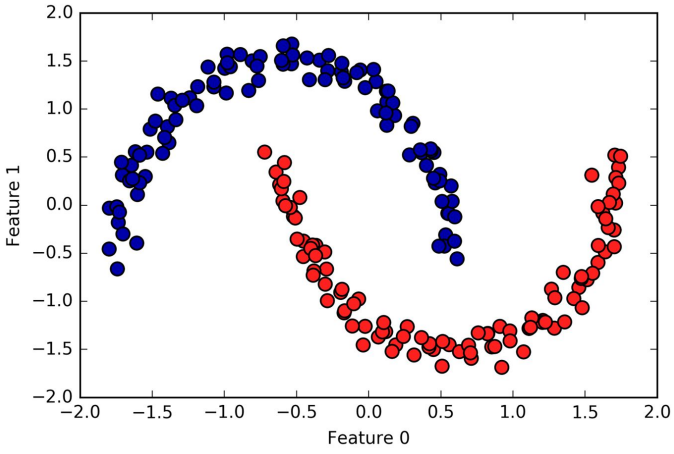
2.2 What are possible ways to speed up training a neural network.

- ■ Using GPUs for the computations.
- ☐ Using drop-out.
- ■ Using batch normalization.
- ■ Initializing the network with weights trained on a similar task.
- ☐ Use batch training instead of using mini-batches.
- ■ Use smaller hidden layers.

2.3 How can you reduce the number of features in a text classification task with a bag-of-word representation?

- ■ Removing uncommon words (min_df).
- ☐ Using higher n-grams (bigrams, trigrams)
- ■ Lemmatization
- ■ Stemming
- ☐ Tf-idf rescaling

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



- ☐ KMeans
- ■ DBSCAN
- ☐ Agglomerative clustering without topology constraint.
- ■ Agglomerative clustering restricted to 3 nearest neighbor graph.

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(optimizer="adam", loss="categorical_crossentropy",
              metrics=["accuracy"])
```

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

3.2 Task: Implement K-Means clustering with random initialization. The algorithm should stop when the total change in the cluster centers has a squared euclidean distance of less than tol. The function euclidean_distances (from sklearn) computes pairwise euclidean distances between two matrices of samples (here it'll return a matrix of shape n_samples x n_clusters).

```
def kmeans(X, n_clusters=10, tol=1e-5):
    init_index = np.random.randint(0, X.shape[0],
                                   size=n_clusters)
    clusters = X[init_index]
    for j in range(X.shape[0]):
        clusters_old = clusters.copy()
        distances = euclidean_distances(X, clusters)
        cluster_labels = np.argmax(distances, axis=1)
        for i in range(n_clusters):
            clusters[i] = X[cluster_labels].mean(axis=0)

        if np.sum((clusters - clusters_old) ** 2) < tol:
            break
    return clusters, cluster_labels
```

Cluster means: axis=1
Cluster_labels needs argmin

Cluster_labels == i
Loop should be "while True"

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 a list of strings in the variable `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(cv.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 explained variance for all components (scree plot), 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], c=y) 2pts
```

5 Concepts (5pt each)

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

5.1 Explain residual neural networks.

5.2 Compute the number of parameters in a densely connected neural network with batch normalization on the MNIST dataset (784 input dimensions, 10 classes) with two hidden layers of dimension 500 each.
784 * 500 (1st layer weights) + 500 (biases)+ 2 * 500 (batch norm)
+500 * 500 (2nd layer weights) + 100 (biases) + 2 * 500 (batch norm)
+ 500 * 10 (output weights) + 10 (output biases)

5.3 Given a function “vec” that computes the word2vec representation of a word, how can we find the word that best matches X in the relation “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 3 reasons why convolutional neural networks are better suited for image recognition than fully connected networks.
Possible reasons:
They can exploit the neighborhood structure of images.
They are invariant to translations.
They require learning less parameters.
The can share parameters between different locations in the image.
They can be used for feature extraction on differently-sized images.