Упражнения по Machine Learning - 2

самостоятельное прорешивание.
ответы сразу после упражнения

Задание 1

1. Загрузите данные affairs_data.csv (из предыдущей лекции), не забудьте закодировать категориальные переменные (get_dummies).
2. Постройте pipeline, который объединяет отбор переменных и обучение SVC с линейным ядром.
3. Последовательно увеличивая количество отобранных переменных (с 1 до 14), обучите построенный pipeline только на отобранных переменных.
4. Постройте график ‘accuracy’ полученных моделей.

Решение 1

1. ```
   from sklearn.pipeline import Pipeline
   selector = SelectKBest(f_classif).fit(X, y)
   my_pipe = Pipeline([('selection', selector), ('learn', SVC(kernel='linear'))])
   means = []
   stds = []
   my_range = xrange(1, 15)
   for k in my_range:
       my_pipe.set_params(selection__k=k)
       scores = cross_val_score(my_pipe, X, y, cv=5, scoring='accuracy')
       means.append(scores.mean())
       stds.append(scores.std())
       print my_pipe.get_params()['selection'].get_support(indices=True)

   plt.errorbar(my_range, means, stds)
   ```

Задание 2. Деплоинг на кластере

1. Задеплоить полученное на семинаре решение на кластере
2. Воспользоваться моделью классификации на основе RandomForest
3. Выбрать пороговое значение, соответсвующее точности 60%
4. Модифицировать скрипт таким образом, чтобы он печатал только пользователей, чья вероятность ответа на маркетинговую кампанию выше выбранного порога

   ```

Решение 2

1. Скрипт sh:
   ```
   #!/bin/bash
   hadoop fs -rm -r -skipTrash marketing_prediction
   hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar \
      -input marketing_data \
      -output marketing_prediction \
      -mapper 'python mapper.py' \
      -file mapper.py \
      -file final_model \
      -file encoders \
      -file onehot \
   ```
2. Сохранить другую модель в файлик по средствам pickle
3. Построить график precision / threshold, выбрать параметр можно визуально
4. Нужно добавить одно условие if перед печатью

Задание 3. L1 feature selection

1. Провести отбор переменных по средствам модели логистической регрессии с l1 регуляризацией, команды:
   ```
   selector = LogisticRegression(penalty='l1', C=0.1)
   selector.fit(X_train, y_train)
   X_train = selector.transform(X_train)
   ```
2. Обучить модель логистической регрессии, используя только отобранные переменные
3. Перебрать различные значения параметра C, выбрать модель, которая на ваш взгляд реализует лучший tradeoff между качеством и сложностью
4. Сохранив дополнительно объект selector, рассчитать модель на кластере (у вас добавится дополнительный шаг -- feature selection, который нужно применять после OneHotEncoding)

Решение 3

1. ---
2. ---
3. selector = LogisticRegression(penalty='l1')
   my_pipe = Pipeline([('selection', selector), ('learn', LogisticRegression())])
   params = [{'selection__C': np.logspace(-5, -1, 5)}]
   clf = GridSearchCV(my_pipe, params, cv=CV, scoring='roc_auc', verbose=3)
   clf.fit(X_train, y_train)
4. ----
