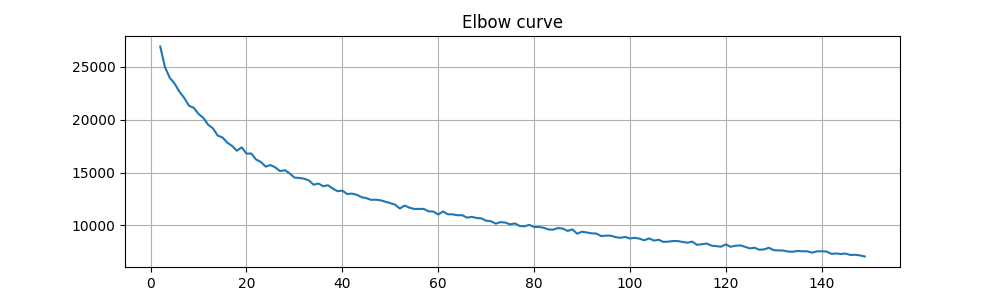
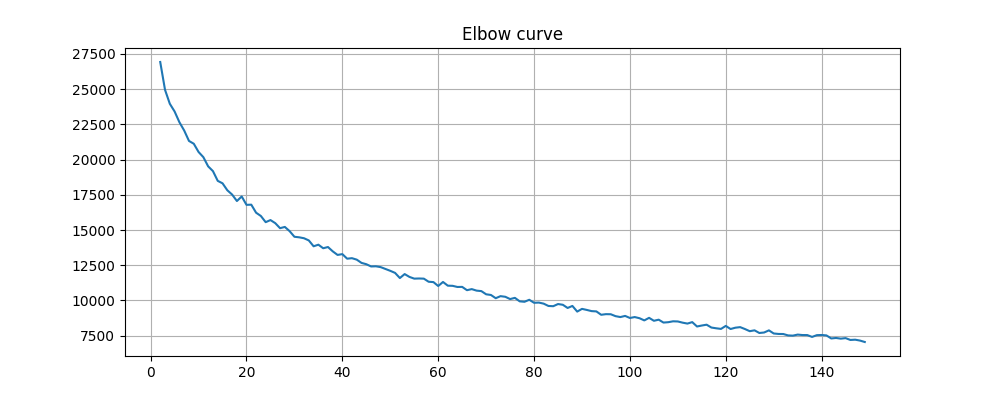
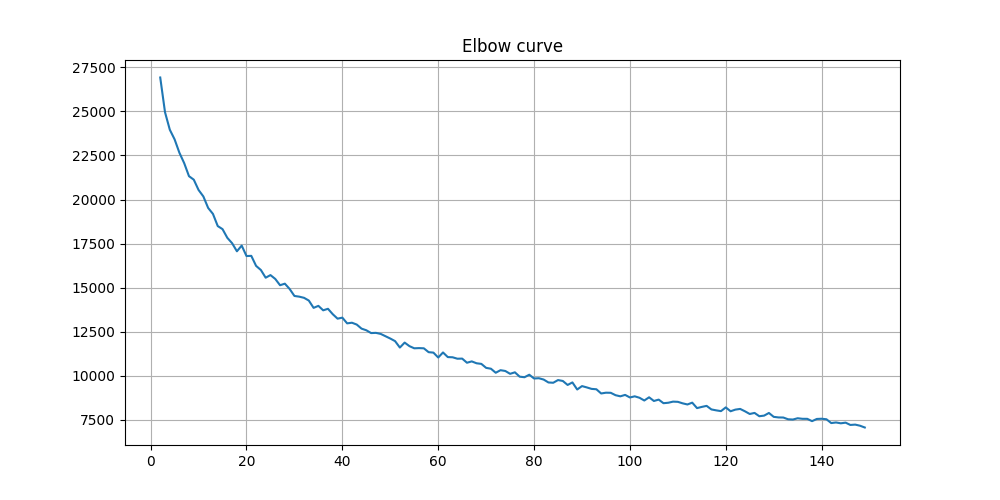
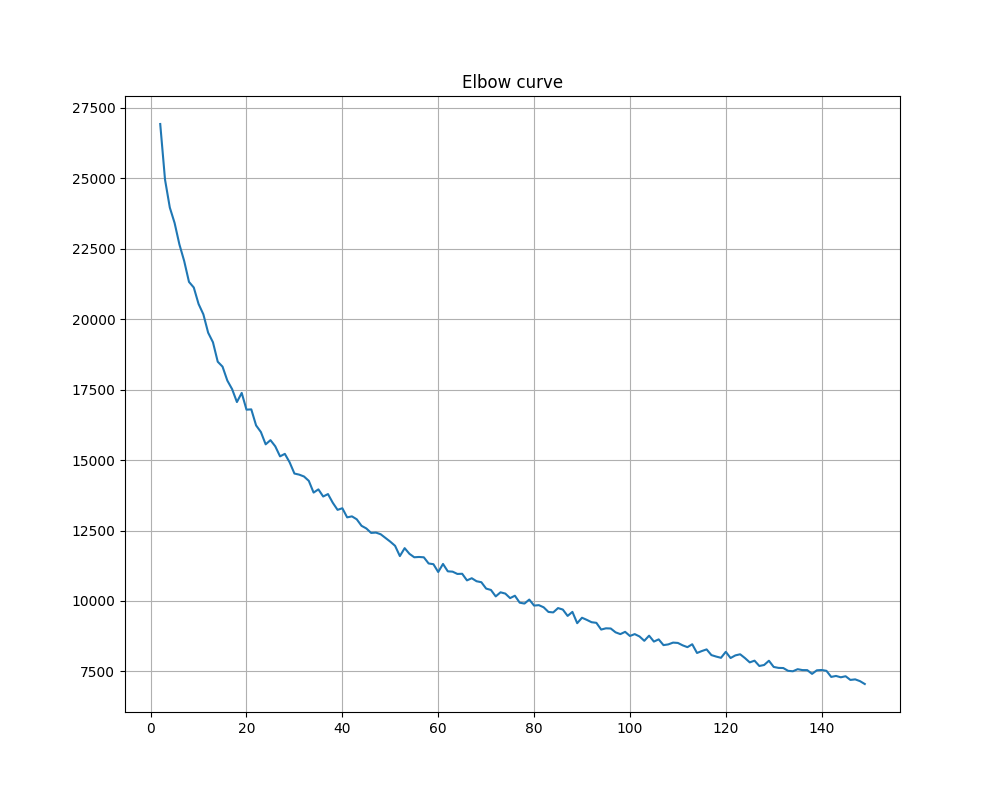
Поиск max\_depth XGBoost  
min = -0.7778341888962922

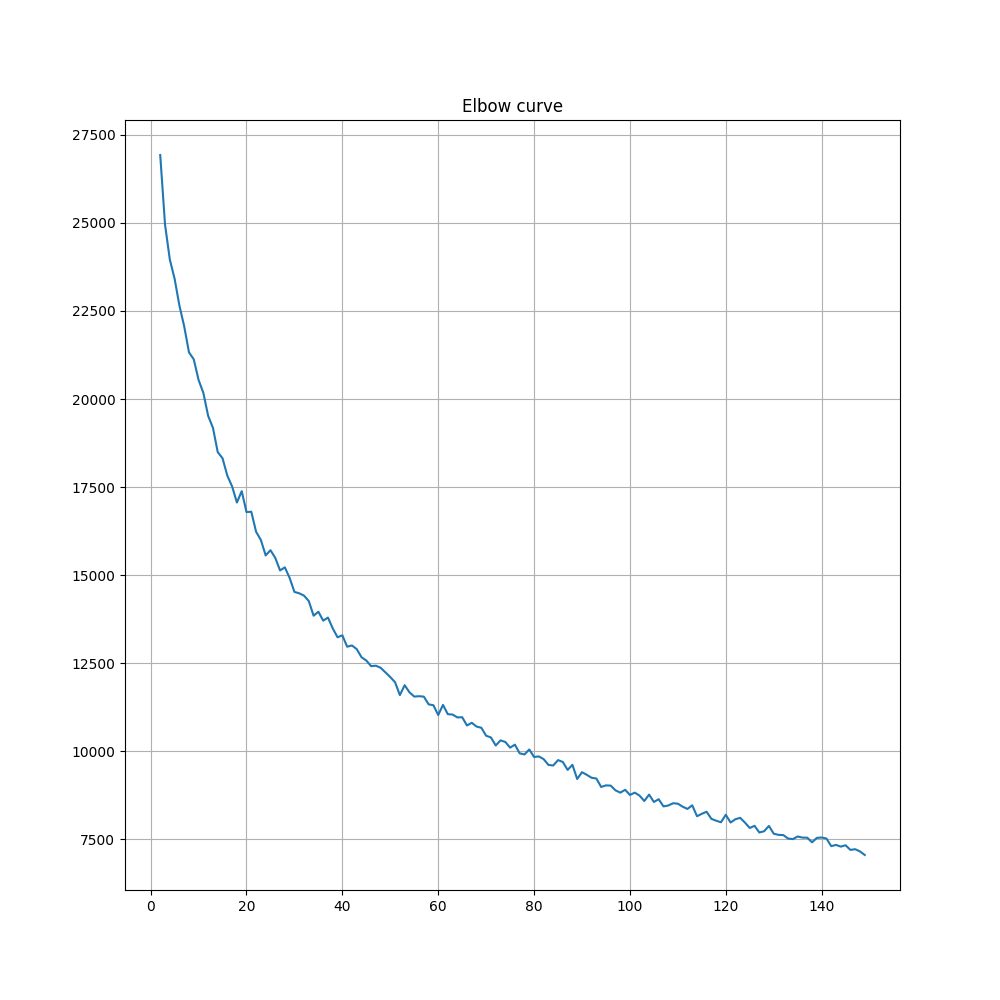
Поиск оптимального кол-во кластеров для тэгов. Картинка 10х8 (предпоследняя), колено = 40











# todo: Начало бустинга XGBoost

# Собираем модель

# %%

import xgboost

from sklearn.model\_selection import cross\_validate

from sklearn.metrics import accuracy\_score

# todo: ksjdkofjsidjfklsjfdjlskdjfkljsdklfjklsjdfkljslkdjfklsjdfkljsldkfjklsj

XGBmodel = xgboost.XGBClassifier(max\_depth=8)

XGBmodel.fit(train\_df, y\_train)

# %%

print(XGBmodel)

# # %%

# # Делаем предикты с помощью XGB модели

# XGBpred = XGBmodel.predict(test\_df)

# XGBpredictions = [round(value) for value in XGBpred]

# %%

# Оценка XBoost-метода ROC-AUC кривой

print('XGBoost ROC-AUC score: ',

roc\_auc\_score(y\_test.values,

XGBmodel.predict\_proba(test\_df)[:, 1]

)

)

# %%

import sklearn.metrics as metrics

probs = XGBmodel.predict\_proba(test\_df)

preds = probs[:,1]

fpr, tpr, threshold = metrics.roc\_curve(y\_test, preds)

roc\_auc = metrics.auc(fpr, tpr)

# %%

plt.title('Receiver Operating Characteristic')

plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc\_auc)

plt.legend(loc = 'lower right')

plt.plot([0, 1], [0, 1],'r--')

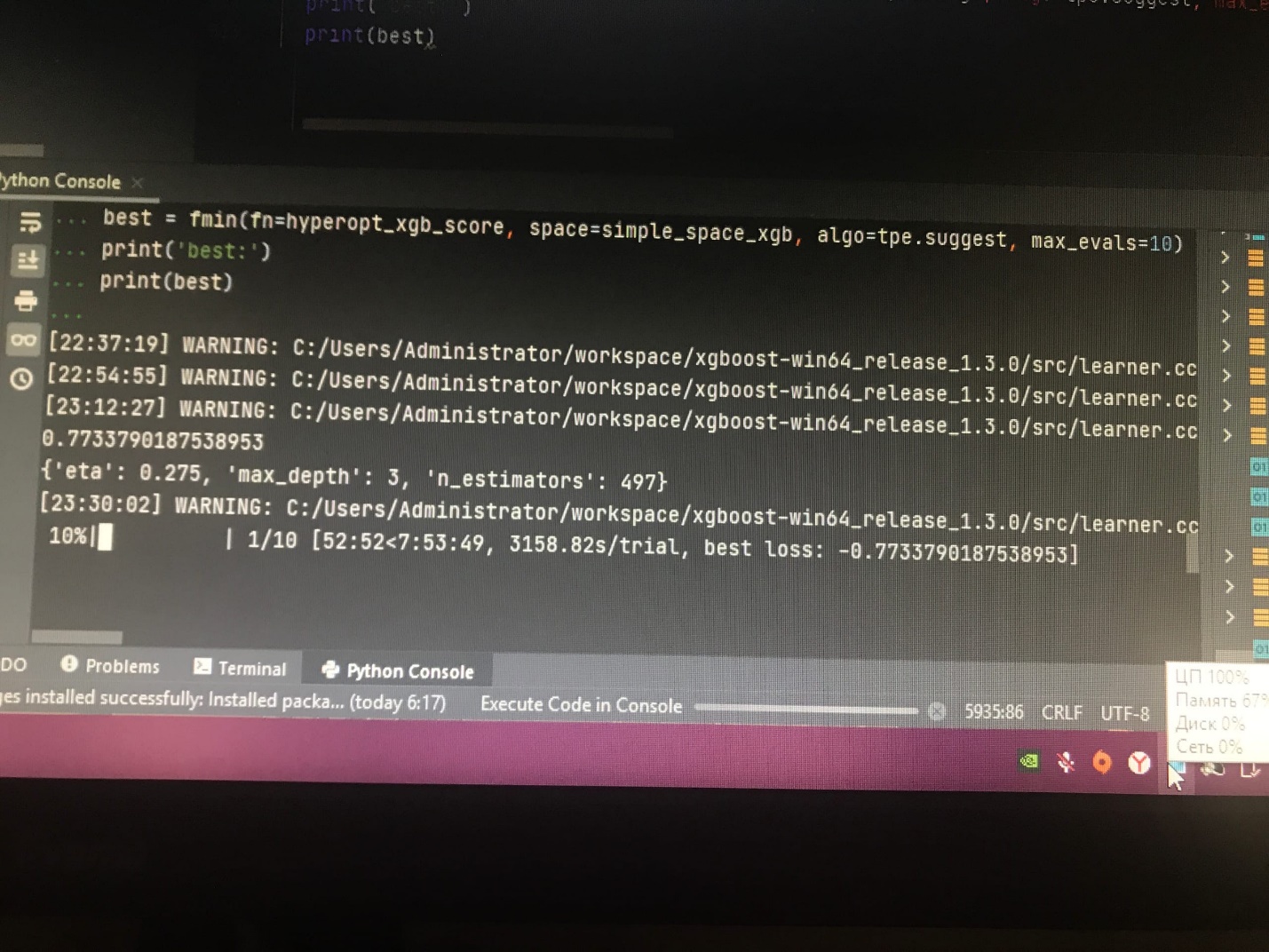
plt.xlim([0, 1])

plt.ylim([0, 1])

plt.ylabel('True Positive Rate')

plt.xlabel('False Positive Rate')

plt.show()



Код оптимизатора:  
